LDOS FILTER PACKAGE - FIRST EDITION

T A B L E O F C O N T E N T S

T H E L D O S F I L T E R P A C K A G E

The LDOS Filter Package will provide you with capabilities not found when operating under the standard LDOS system. It is designed to utilize the device independence which is inherent in the LDOS system.

In describing each filter, the documentation which follows will conform to the syntax used in the LDOS manual. Each filter will be described by a syntax block and an explanation of the function which will be performed by the filter. Examples of the results of the filter will be given where necessary.

Several of the filters will have either two parameters or a counterpart filter which will cause the filter to perform an exactly opposite function. It should be noted here that for all filters, the first filter implemented will be the last one to be executed. For example, if you were to filter the video display (*DO) with LOWER/FLT, and at a later time filter it with UPPER/FLT, realize that LOWER/FLT would take precedence, since it was the first filter implemented. If you wish to re-filter a device with a filter that will perform the opposite function, it is advisable to RESET the device prior to establishing the new filter.

Many of the filters contained in this package will be directed towards devices that are capable of output. Although some of the filters are designed primarily for a specific device, they may be applied to any device meeting the specifications. Please note that more than one device may be filtered with the same filter at the same time. Realize also that each time a device is filtered, a new memory allocation will be used to contain the filter program, regardless of whether or not the device in question was RESET. A global RESET will restore HIGH$ to the top of memory, provided that no background or foreground tasks are active.

NOTE

This filter package is separate from your LDOS system, and will not be supported on the customer service lines. If you have any questions concerning these filters, contact LSI. The number you can use to get support on the filter programs is:

414/241-3066

C A L C / F L T

A keyboard filter to perform Hex/Decimal/Binary conversions and Hex math. The syntax is:

```
============================================================
|  FILTER *KI USING CALC/FLT                               |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

As with all LDOS 5.1 keyboard functions that require the <CLEAR> key, KI/DVR must be set before applying the filter. Once CALC is installed, it is called by pressing the <CLEAR><RIGHT ARROW> keys. If using 5.1.1 or later, the keyboard cannot be in the Extended Character Mode when using CALC. Upon entering the CALC filter, the display will be:

B,C,D,H,M or <BREAK> to abort:

The CALC Filter will normally deal with integer values in the range of 0-65535. If the results of a particular function produce a value outside of this range, CALC will return a value modulo 65535 (i.e. the result will represent a value which has been "wrapped around"). The exception to this will be when using the two's complement value for converting decimal to hex. The CALC options are as follows:

<BREAK> - Aborts to the calling program.

<B> - Converts a Hex or Decimal number to its binary representation. Use the format BHnnnn to convert a Hex number, and BDnnnn to convert a Decimal number. Numbers for conversion must be within the range 0-65535 (x'0000'-x'FFFF').

<C> - Converts a pattern of l's and 0's to a Hex number. Up to 8 binary

digits may be entered. For example, Cl0l010l0 would return X'00AA'.

<D> - Converts a Hex number to Decimal. For example, Dl00 would return 00256. If the Hex number is greater than X'7FFF', both the direct conversion decimal value and the negative two's complement decimal value will be displayed. For example, DFFFF would return both 65535 and -1.

<H> - Converts a Decimal number to Hex. For example, H256 would return X'l00'. Negative values in the range -l to -32768 may also be entered, and will be considered to be two's complement values to be converted. For example, H-1 would return X'FFFF'.

<M> - Allows addition or subtraction of two Hex numbers. The syntax is Mnnnn+nnnn for addition, or Mnnnn-nnnn for subtraction. For example, MF000-1000 would show E000.

LINEFEED/FLT

This filter will either add or remove a linefeed after each carriage return. The syntax is:

```
============================================================
|  FILTER devspec USING LINEFEED/FLT (parm)                |
|                                                          |
|  parameters are ADD or REMOVE                            |
|                                                          |
|  abbr: ADD=A, REMOVE=R                                   |
|                                                          |
============================================================
```

This filter will either add or delete a linefeed after a carriage return. If the parameters are omitted, the message "Parameter Error" will be shown.

The ADD parameter will send a linefeed (X'0A') character to the specified device after every carriage return (X'0D') has been sent.

The REMOVE parameter will check the character to be sent after a carriage return, and will not send it if it is a linefeed.

The LINEFEED filter's primary uses will probably be as a line printer or Comm Line filter.

LISTBAS/FLT

A filter which separates packed BASIC programs (statements separated by colons) into individual statements and expressions. The syntax is:

```
============================================================
|  FILTER devspec USING LISTBAS/FLT                        |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

This filter may be used with the *DO (video) or the *PR (printer) devices. It is used to restructure the appearance of BASIC programs that contain multiple statements on one logical line. When BASIC programs are written in this manner, they are referred to as "packed" and are very difficult to work on. When the LISTBAS filter is installed, each BASIC line will be output in a more structured manner.

The effects of this filter are limited to reformatting each line of program that passes through it. It is important to note that the actual BASIC program is in NOT CHANGED IN ANY WAY when using this filter.

The following is a sample "packed" BASIC line:

```
10FORX=32TO64:PRINTX;CHR$(X);CHR$(INT(X+( Q*64)/7.5)):FORY=0TO200:NEXTY:NEXTX
```

Filtering *PR with LISTBAS/FLT and performing an LLIST of the above line would produce the following output on the line printer:

```
100 FORX=32T064:
      PRINTX;
      CHR$ (x) ;
      CHR$ (INT (X+ ( Q*64) /7.5) ):
       FORY=0TO200:
        NEXTY:
         NEXTX
```

The same results will be obtained if you filter *DO with LISTBAS and then LIST this line in BASIC.

A linefeed will follow each additional statement (one following a colon ":") encountered in a BASIC text line. The first statement in a line will follow the line number, and a space will always separate the line number from the first statement in a line. The first character in the second statement of each line will always appear in column 7. Subsequent statements in each line will be tabbed over one space from the preceding statement (i.e. the first character of the third statement will appear in column 8).

Certain delimiters which are not contained within a literal string (i.e. inside the quote marks of a PRINT statement) are also acted upon by the LISTBAS filter. Each expression encountered after a semicolon ";" will be printed on a new line, starting in the same column as the preceding line.

A space will always precede the opening parenthesis denoting an argument, subscript, or expression, and a space will always follow the closing parenthesis. (Because commas serve various functions in LBASIC, they will be ignored by the filter, and will not produce the same results as the use of semicolons in PRINT statements.)

In some cases, the length of a statement may cause characters to be displayed beyond the last available column of the device in question. If this is the case, any characters to be printed beyond the last available column will be "wrapped around" to the first available column of the next line.

Please note that due to ROM interpreting, imbedded line feed characters (X'0A' in a logical program line) will be treated as colons and terminate that BASIC text line as far as formatting it is concerned. Formatting will begin again in the next printed line as though it were the next logical line. When an imbedded X'0A' is encountered, all of the line will be printed correctly but the format will not be totally correct, as the formatting will start over at the left margin as though a new logical line were encountered.

When filtering the Video Display (*DO) with LISTBAS/FLT, the user's program will be formatted automatically when it is entered from the keyboard. Although this has its advantages, it also has its drawbacks. The major one is found in the use of the <BACKSPACE> key. The video will display the program "formatted" as it is being entered and will apply spaces and linefeeds where appropriate. When the programmer executes a backspace, the last character that was typed will be removed from the BASIC line in memory. This occurrence may not be readily apparent to the user, as BASIC is working on the line in memory, but what is appearing on the video is what was modified for display by the filter. So, be careful when using this filter during writing or editing your programs, unless you thoroughly understand its operation. It is for this reason we recommend that most users use this filter on the *PR device only.

It should also be noted that while the filter is in place on the *DO or *PR devices, execution of BASIC programs which PRINT or LPRINT colons, semicolons, or parenthesis will be formatted like the BASIC text lines. To solve this, simply turn off the filter by using the BASIC CMD command (CMD"RESET *DO" or *PR). Keep in mind though that this makes it impossible to recover LISTBAS in BASIC. The user would need to return to LDOS, FILTER devspec, re-execute LBASIC, and then load the desired BASIC program.

When using LISTBAS/FLT, it may be necessary to install PR/FLT first if your line printer can not respond to a line feed (X'0A').

The next two pages will display sample listings of a BASIC program. The first example will show a program listing in its "packed" form, while the second example will illustrate a listing of the same program lines with LISTBAS/FLT established. Note that the device used for these examples was *DO.

The following is a listing of a packed BASIC program without the LISTBAS filter :

```
1 'INVMENU V2.0
2 GOTO 5010
5 CLS:PRINT@464,"NOW LOADING MODULE":RETURN
200 IF F<1 OR F>255 THEN IN$="":RETURN ELSE PRINT@TA,"";
210 IN$=STRING$(F,32):XX=USR9(VARPTR(IN$)):RETURN
600 OPEN"I",4,"INVINDEX:0":CLOSE4:OPEN"R",4,"INVINDEX":FOR L=1 TO
128:FIELD4,((L-1)*2)AS DM$,2ASPZ(L):NEXTL:RETURN
605 ONERRORGOTO0:END
1000 NF$="INVDATA"+MID$(STR$(N),2)+":"+MID$(STR$(N),2):RETURN
1100 FIELDN,3 AS DM,11 AS DI,5 AS DX,8 AS DG,24 AS DM,2 AS DK,3 AS DL,200 AS
DM:RETURN
3000 CLS:PRINTTAB(15)"GALACTIC SOFTWARE INVENTORY MASTER":PRINT:
PRINTTAB(15)"*****   M A I N    M E N U   *****"
3005 PRINT:PRINT"******************  W A R N I N G  ******************"
PRINTTAB(3)"DISKS MAY  ** N 0 T **  BE REMOVED UNTIL THE END OF SESSION"
3010 PRINT:PRINTTAB(7)"<S> - SEARCH/MODIFY/ADD INVENTORY ITEMS":PRINT
TAB(7)"<O> - ORDER PLACEMENT/EDIT/PROCESS & DAILY INPUT/PROCESS":PRINT
TAB(7)"<R> - REPORT GENERATOR"
3015 PRINTTAB(7)"<M> - MONTHLY/QUARTERLY/YEARLY PROCESSING":PRINTTAB(7)"<@> -
END OF SESSION":PRINT:PRINTTAB(7)"ENTER A SELECTION"
3020 TA=924:F=2:GOSUB200
3025 IF IN$="S" THEN GOSUB5:RUN"INVENT:0"
3026 IF IN$="O" AND LT>0 THEN GOSUB5:RUN"INVORDER:0"
3027 IF IN$="R" AND LT>0 THEN GOSUB5:RUN"INVREPOR:0"
3028 IF IN$="M" AND LT>0 THEN 3400 ELSE IF IN$<>"@" THEN 3020
```

Here is the same program with LISTBAS filter established:

```
1 'INVMENU V2.0
2 GOTO 5010
5 CLS:
      PRINT@464,"NOW LOADING MODULE":
       RETURN
200 IF F<1 OR F>255 THEN IN$="":
      RETURN ELSE PRINT@TA,"";
210 IN$=STRING$ (F,32) :
      XX=USR9 (VARPTR (IN$) ) :
       RETURN
600 OPEN"I",4,"INVINDEX:0":
      CLOSE4:
      OPEN"R" ,4,"INVINDEX":
       FOR L=1 TO 128:
        FIELD4, ( (L-l) *2) AS DM$,2ASPZ (L) :
         NEXTL:
          RETURN
605 ONERRORGOTO0:
      END
1000 NF$="INVDATA"+MID$ (STR$ (N) ,2) +":"+MID$ (STR$ (N) ,2) :
      RETURN
1100 FIELDN,3 AS DM,11 AS DI,5 AS DX,8  AS DG,24 AS DM,2 AS DK,3 AS DL,200 AS
DM:
      RETURN
3000 CLS:
       PRINTTAB (15) "GALACTIC SOFTWARE INVENTORY MASTER":
        PRINT:
         PRINTTAB (15) "*****   M A I N    M E N U   *****"
3005 PRINT:
      PRINT"******************  W A R N I N G  ******************":
       PRINTTAB (3) "DISKS MAY  ** N 0 T **  BE REMOVED UNTIL THE END OF
SESSION"
3010 PRINT:
      PRINTTAB (7) "<S> - SEARCH/MODIFY/ADD INVENTORY ITEMS":
       PRINTTAB (7) "<O> - ORDER PLACEMENT/EDIT/PROCESS & DAILY INPUT/PROCESS":
        PRINTTAB (7) "<R> - REPORT GENERATOR"
3015 PRINTTAB (7) "<M> - MONTHLY/QUARTERLY/YEARLY PROCESSING":
      PRINTTAB (7) "<@> - END OF SESSION":
       PRINT:
        PRINTTAB (7) "ENTER A SELECTION"
3020 TA=924:
      F=2:
       GOSUB200
3025 IF IN$="S" THEN GOSUB5:
      RUN"INVENT:0"
3026 IF IN$="O" AND LT>0 THEN GOSUB5:
      RUN"INVORDER:0"
3027 IF IN$="R" AND LT>0 THEN GOSUB5:
      RUN"INVREPOR:0"
3028 IF IN$="M" AND LT>0 THEN 3400 ELSE IF IN$<>"@" THEN 3020
```

L0WER/FLT and UPPER/FLT

These two filters convert alphabetic characters in the range A-Z to either all UPPER or all lower case. The syntax is:

```
==============================================================
|  FILTER devspec USING LOWER/FLT                            |
|  FILTER devspec USING UPPER/FLT                            |
|                                                            |
|  no parameters are required                                |
|                                                            |
==============================================================
```

The UPPER filter will convert all characters in the range (a-z) to UPPER case. The LOWER filter will convert all characters in the range (A-Z) to lower case. Either filter may be applied to any device capable of output.

MONITOR/FLT

This filter will monitor a specified device and display special symbols for all non-ASCII characters. The syntax is:

```
============================================================
| FILTER devspec USING MONITOR/FLT                         |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

The Monitor filter can be used on any device capable of output, and will display non-ASCII characters in the following manner:

Any character less than a space (X'20') will be displayed as a percent sign (%) followed by an ASCII representation of the character value + X'41'. Thus an X'0D' would be displayed as %N, an X'0A' as a %K, etc.

Any character greater than an X'7F' will be displayed as the character X'5B'. This will be either an <UP ARROW> or a <LEFT BRACKET>, depending on your computer and/or your type of output device (printer, terminal, etc).

This filter is primarily useful when it is suspected that control characters not normally visible are being sent to a device, or are present in some type of ASCII file. You will be able to determine the exact character using this filter.

Realize that for certain devices (namely *PR), applying the MONITOR filter may produce extra characters in a line of display, and thus may affect character and/or line counters.

PAGEPAWS/FLT

This filter (pronounced PAGE PAUSE) will cause a pause after every Top-of-Form character, and will wait until <ENTER> is pressed to continue. The syntax is:

```
============================================================
| FILTER devspec USING PAGEPAWS/FLT                        |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

Although this filter can be applied to any device capable of output, it will primarily be useful with line printers that use single sheet paper. Once this filter is applied, any Top-of-Form character (X'~C') sent to the specified device will cause the system to pause until the <ENTER> key is pressed. This will allow a new sheet of paper to be inserted into the printer.

REM0VE UTILITY PR0GRAM (REMOVE/CND)

The REMOVE program is provided to remove a specific character value from a file. It could be used to remove the character {X'00'} from a file to be loaded into SCRIPSIT. It could also be used to remove line feeds {X'0A'} from a CPM assembler source file converted to LDOS. It could be used...

The syntax is:

```
==============================================================
|  REMOVE filespec1 filespec2 ( BYTE=parm)                   |
|  REMOVE filespec1 partspec  ( BYTE=parm)                   |
|                                                            |
|  parm entry is described below.                            |
|                                                            |
|  BYTE=ddd                                                  |
|  BYTE=X'hh' or BYTE=X'hh                                   |
|  BYTE="c"                                                  |
|          Will strip the character identified as "parm".    |
|          ddd, hh, c = the character code to strip.         |
|                                                            |
|  abbr: BYTE=B                                              |
|                                                            |
==============================================================
```

The REMOVE operation is essentially a COPY function of the original file to a new file. Any character matching up with the designated "remove" character is not copied over to the new file.

The source file and the destination file must be different files. REMOVE will abort the operation if you enter identical file specifications. You may omit the file extension if the file has an extension of /TXT. The default extension of TXT will be assumed if none is provided.

When REMOVE parses the command line entered, the FILENAME, EXTension, PASSWORD, and DRIVESPEC for the destination file will automatically default to those of filespec1 if they are not specified. See the following examples.

EXAMPLES of REMOVE: filespec1 filespec2

```
REMOVE MNET1015/TXT:7 MNET1015/SCR:7 (BYTE=0)
REMOVE MNET1015:7 /SCR
```

These commands will copy the file "MNET1015/TXT:7" to "MNET1015/SCR:7" while removing all NULL characters.

```
REMOVE STRANGE:1 DOUBLEO:7 (BYTE=90)
REMOVE STRANGE:1 DOUBLEO:7 (BYTE=X'5A')
REMOVE STRANGE:1 DOUBLEO:7 (B="Z")
```

These commands will remove all appearances of the letter "Z" while transferring the file "STRANGE/TXT:1" to "DOUBLEO/TXT:7".

SLASH0/FLT

This filter will allow a printer capable of backspacing to print a slashed zero (0) if that character is not a normal part of its character set. The syntax is:

```
============================================================
|  FILTER *PR USING SLASH0/FLT                             |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

This filter is intended for use with printers that can act on a backspace character (X'08'), such as daisy wheel printers. It will cause the printer to backspace and print a "/" character every time a zero is printed. This will create a slashed zero character (0).

STRIP7/FLT

This filter will strip the high bit off of each character passed through the filter. The syntax is:

```
============================================================
|  FILTER devspec USING STRIP7/FLT                         |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

This filter will strip the high bit off of each character filtered, in effect converting any characters outside the ASCII range to ASCII. It may be applied to any device capable of output.

STRIPCNT/FLT

This filter will replace any character above X'7F' or below X'20' with a pound sign (#). The syntax is:

```
============================================================
|  FILTER devspec USING STRIPCNT/FLT                       |
|                                                          |
|  no parameters are required                              |
|                                                          |
============================================================
```

This filter may be applied to any device capable of output. It was designed to keep unwanted control characters and space compression characters from being sent to a device not capable of accepting them.

For example, you may wish to use the LIST Library command to examine a data file that contains both ASCII and non-ASCII characters. This filter will prevent unwanted characters from destroying the format of the video display.

Carriage returns (X'0D'), linefeeds (X'0A'), and tab (X'09') characters will be left unchanged by this filter.

TITLE/FLT

This filter will allow titling of printed output. NOTE: it is for use only with a printer (normally *PR). The syntax is:

```
============================================================
|  FILTER *PR USING TITLE/FLT ( TITLE="title",D=switch)     |
|                                                           |
|  TITLE    is the user entered title string. It can be up  |
|           to 20 characters long.                          |
|                                                           |
|  D=       the switch YES or NO. If D=YES, the date and    |
|           time strings will be printed after the title.   |
|           The default is D=YES.                           |
|                                                           |
|  abbr: YES=Y, NO=N                                        |
|                                                           |
============================================================
```

This filter will allow a user entered title to be printed after every Top of Form (X'0C') character is sent to the line printer. The D parameter will print the Date and Time strings after the title string in the following format:

TITLE DD/DD/DD TT:TT:TT

The title string may be in upper and lower case. If a title longer than 20 characters is entered, all characters after the 20th will be ignored. If no title is entered, the message "Parameter error" will be displayed, and the filter will not be installed.

The filter will perform a Top-of-Form and print the title string immediately after it has been applied.

TRAP/FLT

This filter will trap and throw away a specified character each time the character tries to go through the filter. The syntax is:

```
============================================================
|  FILTER devspec USING TRAP/FLT ( CHAR=X'hh')              |
|  FILTER devspec USING TRAP/FLT ( CHAR=dd)                 |
|                                                           |
|  X'hh   represents the desired character in hexadecimal.  |
|    dd   represents the desired character in decimal.      |
|                                                           |
|  abbr:  CHAR=C                                            |
|                                                           |
============================================================
```

This filter may be used with any device capable of output. For example, the Cursor On character (X'0E') causes many printers to either start underlining or go into the expanded print mode. Before issuing a LINK *DO *PR command, you may wish to filter *PR with TRAP to eliminate all X'0E' characters.

Another use might be when using the RS-232 driver and the Comm Line (*CL). Filter *CL with TRAP to prevent any X'17' characters from switching the video into the 32 cpl mode.

It is possible to trap for more than one character at any given time. To do so, apply the TRAP filter as many times as the number of characters you wish to trap for. Realize that every time the TRAP filter is implemented, an additional high memory allocation will be used.

TRANSLATI0N FILTER PR0GRAM (XLATE/FLT)

The FILTER program XLATE/FLT is provided to enable a code translation within the INPUT/OUTPUT path of any device so as to adapt to non-ASCII peripherals or apparatus requiring special codes. The syntax is:

```
============================================================
|  FILTER devspec XLATE/FLT filespec (parm,parm,...)       |
|                                                          |
|  filespec    the ASCII file which contains the           |
|              translation table to be used. The default   |
|              extension is /XLT.                          |
|                                                          |
|  parms are the parameters described below.               |
|                                                          |
|  INPUT   Will force INPUT only translation in a device   |
|          capable of INPUT and OUTPUT.                    |
|                                                          |
|  NULL    Provides NULL as INPUT code translation for     |
|          each output code translated (see details).      |
|                                                          |
|  OUTPUT  Will force OUTPUT only translation in a device  |
|          capable of INPUT and OUTPUT                     |
|                                                          |
|  PRIME=ddd                                               |
|  PRIME=X'hh' or PRIME=X'hh                               |
|  PRIME="c" or PRIME="c                                   |
|          Will force initialization of all table codes.   |
|          ddd, hh and cc represent the initialization     |
|          character code to be used.                      |
|                                                          |
|  all parameters can be abbreviated to their first letter |
============================================================
```

XLATE/FLT is used to FILTER the device specified, according to translation data stored in filespec (an ASCII translation table file). This FILTER program adds certain enhancements to the entire operation of LDOS. By providing the user with an easy means of translating data in an I/O channel, greater versatility in "talking" to peripheral equipment is achieved.

Included on the LDOS Filter disk are two translation table files - EBCDIC/XLT and DVORAK/XLT. The EBCDIC/XLT translation table will allow for an ASCII to EBCDIC / EBCDIC to ASCII translation of data sent to and received from I/O channels. This may be useful to communicate with an EBCDIC mainframe. The DVORAK/XLT translation table will allow you to convert your keyboard to a DVORAK keyboard.

The user may also create his own translation tables to be used with XLATE/FLT. For example, printers containing special character sets can be supported by translating unused codes to the special ones. Other uses for translation, such as trapping specific codes in an I/O channel, will be realized as you become more experienced with the function of filtering.

The following pages will detail the use and installation of the XLATE/FLT. At the end of this section you will find a listing of the translation tables EBCDIC/XLT and DVORAK/XLT.

Technical Note: All device output originated with the @CTL system call will be passed without translation.

TRANSLATE DATA FILE DETAILS

The translation table specified in the command line as "filespec" contains the translation codes to be used by XLATE/FLT. If the file extension is omitted, a default extension of "/XLT" will be used. The default extension will aid in the identification of your translation table files and is recommended for use. The translation data is entered into the file as free-form translate fields. Each field is of the syntax:

```
aa=bb
```

where "aa" is equal to the OUTPUT code value prior to translation and "bb" is equal to the OUTPUT code after translation by XLATE/FLT. Conversely, "bb" is also taken to be equal to the INPUT code value prior to translation and "aa" is equal to the INPUT code value after translation. Thus, by entering only one pair of data values, both I/O directions are handled. If your needs require an INPUT translation that is not the reverse image of the OUTPUT translation, you will have to establish an INPUT translation table, using the parameter INPUT, and an OUTPUT translation table, using the parameter OUTPUT.

The values "aa" and "bb" can be entered either as two hexadecimal digits (such as lA, 2f, 0l...) or as a single character contained in double quotes (such as "A", "#", "z"...). The format of a field can be any of the following four types:

```
oo=ii
"o"=ii
oo="i"
"o"="i"
```

The closing double-quote may be omitted if desired to save key stroke entries. Fields are separated from other fields by use of one or more spaces and/or carriage returns. Spaces and carriage returns will serve primarily as delimiters for the data in the translation table. They can be freely used in the translation table to provide a more readable or logically connected set of translation fields.

If the first character of a field is a period ("."), the character string starting from the period up to the next carriage return will be treated as a comment and will NOT be interpreted as translate data. These comments may be used for documenting your translation table files.

The translation table may be generated by using the BUILD library command, or by using the LDOS text editor, LED. Any other word processor may also be used, provided the data file generated is a pure ASCII file. If you are using SCRIPSIT (tm), it is not necessary to save your text with the ",A" option.

In order to perform a code translation in an I/O channel, the translate data file must contain a field entry for every code in the code set. For example, your base machine is an ASCII machine which has a code set of 128 distinct "characters" and you are going to translate the *CL device to EBCDIC. Therefore your translate data file will contain 128 ASCII codes translated to their EBCDIC equivalent. Table II is such a table and can be used to perform the necessary translation. Since EBCDIC has 256 distinct code values, what happens to the other 128 table positions? The codes in the range 0-255 not appearing in the translate data file will take on values dependent on the parameters PRIME and NULL. Also, if the PRIME parameter is not used, then all code values with an identical translated value need not appear in the translate data file as the INP and OUT tables will be initialized to a one-to-one translation (i.e. 00=00, 01=01, 02=02...).

Two-way devices will be filtered in both directions unless over-ridden by the INPUT or OUTPUT parameters (INPUT and OUTPUT are mutually exclusive). A one-way INPUT device will be automatically established with only an INPUT filter table. A one-way OUTPUT device will be automatically established with only an OUTPUT filter table. An attempt to filter a device that has neither INPUT nor OUTPUT capability will result in the display of the error message:

Device has no I/O capabilities!

Restricting the filtering to a single direction will reduce the length of the resident filter program by 256 bytes. The device direction can be ascertained by using the DEVICE Library command and noting the direction of the arrows following the device name.

=> implies one-way OUTPUT
<= implies one-way INPUT
<=> implies two-way INPUT and OUTPUT

The direction can also be ascertained by analyzing the TYPE byte of the Device Control Block (DCB). As noted in the Technical Information - Device Control Block, Page 1; DCB+0, bit 0 if set to a 1 implies INPUT capability; DCB+0, bit 1 if set to a 1 implies OUTPUT capability.

PARAMETER DETAILS

INPUT

If you want to provide one-way only translation in the INPUT direction of a two-way device channel, specify this parameter. The filter program will omit the OUTPUT translate table and reduce the length of the resident filter program by 256 bytes. If this parameter is used and the device does not have INPUT capability, then the error message:

Device has no INPUT capabilities!

will be displayed.

OUTPUT

If you want to provide one-way only translation in the OUTPUT direction of a two-way device channel, specify this parameter. The filter program will omit the INPUT translate table and reduce the length of the resident filter program by 256 bytes. If this parameter is used and the device does not have OUTPUT capability, then the error message:

Device has no OUTPUT capabilities!

will be displayed.

PRIME

The INPUT and OUTPUT tables are normally initialized with:

00=00 01=01 02=02 ... FD=FD FE=FE FF=FF

so that all codes not entered into the translate data file will be interpreted as one-to-one translation (i.e. they will be passed as if no translation was in effect). This may not be the desired effect. You may want all codes not listed in the translate data file to be completely ignored. A NULL is usually ignored in most systems. Thus by specifying:

FILTER devspec XLATE filespec (PRIME=0)

both translate tables will be initialized to NULL prior to reading the data file. If you want some indication of a code not translated but passed, then assign some seldom-used character for the code, such as a tilde, and prime the tables with:

FILTER devspec XLATE filespec (PRIME=X'7E')

Perhaps you want to assign a keyboard generatable character. PRIME also accepts the argument in string form. For instance, suppose you want to define all untranslated codes as a question mark. You could enter:

FILTER devspec XLATE filespec (PRIME="?")

All positions of the INPUT and OUTPUT tables not located in the translate data file will be translated to the "?" character.

NULL

The NULL parameter is more easily demonstrated than defined. Since the INPUT translation is defined by the reverse image of the OUTPUT translation data, cases could arise where the reverse image does not constitute all codes identified in the OUTPUT direction. You may, in fact, want to ensure that a code value received after translation arises from one and only one value before translation. Consider the following simplified translate data file:

13=03 0D=0A 0D=0D

The first field will translate an OUTPUT Control-S (X-OFF) to a Control-C and translate an INPUT Control-C to a Control-S. Note that if a Control-S is also INPUT, it would be one-to-one translated to a Control-S. Since this would NOT be acceptable, the NULL parameter is entered. By using NULL, the translation filter will perform three operations when parsing the field, 13=03. First, an OUTPUT X'13' is translated to an X'03'. Second, an INPUT X'03' is translated to an X'13'. Third, an INPUT X'13' is translated to an X'00'. The same result could have been accomplished without the use of NULL by entering, into the translate data file, two additional fields of:

```
00=l3 00=00
```

which would have forced INPUT translation of X'13' to NULL with the '00=00' supplied to ensure that OUTPUT X'00' will not be translated to X'13'.

The second field was used to translate INPUT line feeds to carriage returns. Unfortunately it would also translate OUTPUT carriage returns to line feeds - not what is wanted. The third field corrects for this deficiency by translating INPUT/OUTPUT carriage returns to carriage returns (a one-to-one identity translation).

If you use the EBCDIC translation data file and want to ensure that only file-entered codes are passed, then use the PRIME=0 and NULL parameters - or enter the remaining 128 EBCDIC codes that have no ASCII correlation.

ERROR MESSAGES

1. Device has no I/O capabilities!

An attempt was made to XLATE a device that supported neither INPUT nor OUTPUT.

2. Device has no INPUT capabilities!

An attempt was made to XLATE a device that did NOT support INPUT capability and you specified the INPUT parameter.

3. Device has no OUTPUT capabilities!

An attempt was made to XLATE a device that did NOT support OUTPUT capability and you specified the OUTPUT parameter.

4. Parameter Error - Translation aborted!

An erroneous parameter was entered or one not conforming to the standard LDOS syntax.

5. Translate data syntax error - Missing "="!

The OUTPUT character of a translation data pair was not followed by an equal sign.

6. Odd # of hex digits!

Only one hex digit was found in a translate data pair.

7. Bad hex digit encountered!

A character not in the range 0-9, A-F, or a-f was found in a hexadecimal translate data field.

8. Missing translate data hex pair!

A translate data field was found that had no post-translate data.

9. Translate data file too big!

The size of the data file exceeds the memory available to store it. Reduce unnecessary comments and multiple spaces. The translate data file must be 3K or less.

DVORAK/XLT - Table I

```
. ASCII from DVORAK Translation Data File
.     This filter implements a DVORAK keyboard as explained
.     in December 1980 issue of "80 MICROCOMPUTING"
"*"="!" "!"="*" "S"="+" "W"="," "V"="." "Z"="/" "0"="6" ":"="1"
"7"="2" "5"="3" "3"="4" "1"="5" "9"="6" "0"="7" "2"="8" "4"="9"
"8"=":" "S"=";" "W"="<" "V"=">" "Z"="?" "X"="B" "J"="C" "E"="D"
">"="E" "U"="F" "I"="G" "D"="H" "C"="I" "H"="J" "T"="K" "N"="L"
"B"="N" "R"="O" "L"="P" "?"="Q" "P"="R" "O"="S" "Y"="T" "G"="U"
"K"="V" "<"="W" "Q"="X" "F"="Y" ";"="Z" " x"="b" "j"="c" "e"="d"
"."="e" "u"="f" "i"="g" "d"="h" "c"="i" "h"="j" "t"="k" "n"="l"
"b"="n" "r"="o" "l"="p" "/"="q" "p"="r" "o"="s" "y"="t" "g"="u"
"k"="v" ","="w" "q"="x" "f"="y" "+"="z"
```

EBCDIC/XLT - Table II

```
. ASCII to EBCDIC Translate Filter data
00=00 01=01 02=02 03=03 04=37 05=2D 06=2E 07=2F
08=16 09=05 0A=25 0B=0B 0C=0C 0D=0D 0E=0E 0F=0F
10=10 11=11 12=12 13=13 14=3C 15=3D 16=32 17=26
18=18 19=19 1A=3F 1B=27 1C=1C 1D=1D 1E=1E 1F=1F
20=40 "!"=5A  22=7F "#"=7B "$"=5B "%"=6C "&"=50
"'"=7D "("=4D ")"=5D "*"=5C "+"=4E ","=6B "."=4B "/"=61
"0"=F0 "1"=F1 "2"=F2 "3"=F3 "4"=F4 "5"=F5 "6"=F6 "7"=F7
"8"=F8 "9"=F9 ":"=7A ";"=5E "<"=4C "="=7E ">"=6E "?"=6F
"@"=76 "A"=C1 "B"=C2 "C"=C3 "D"=C4 "E"=C5 "F"=C6 "G"=C7
"H"=C8 "I"=C9 "J"=D1 "K"=D2 "L"=D3 "H"=D4 "N"=D5 "O"=D6
"P"=D7 "Q"=D8 "R"=D9 "S"=E2 "T"=E3 "U"=E4 "V"=E5 "W"=E6
"X"=E7 "Y"=E8 "Z"=E9  5B=AD  5C=E0  5D=BD  5E=5F  5F=6D
 60=79 "a"=81 "b"=82 "c"=83 "d"=84 "e"=85 "f"=86 "g"=87
"h"=88 "i"=89 "j"=91 "k"=92 "l"=93 "m"=94 "n"=95 "o"=96
"p"=97 "q"=98 "r"=99 " s"=A2 "t"=A3 "u"=A4 "v"=A5 "w"=A6
"x"=A7 "y"=A8 "z"=A9  7B=8B  7C=6A  7D=9B  7E=A1 7F=07
```

I M P O R T A N T N O T I C E

The LDOS FILTER package is a product of Logical Systems, Incorporated, and has been designed and tested to work with the LDOS (TM) operating system, a product of Logical Systems, Incorporated.

This product is sold on an "as is" basis. Logical Systems, Incorporated makes no expressed or implied warranty of any kind with regard to the software or documentation. Under no circumstances will Logical Systems, Incorporated assume any liability for incidental or consequential damages resulting from the use of this product.

From time to time, updates or enhancements to this product may become available for a nominal charge. Customer Service information on this product and any available updates may be acquired by contacting Logical Systems, Incorporated at the following address:

Logical Systems, Incorporated
11520 N. Port Washington Rd.
Mequon, WI 53092

(414) 241-3066