1 of 7

71-00002 PROGRAM DESCRIPTION

Program Title ROWCOL

Contributor Hewlett-Packard Company

Address 1000 NE Circle Blvd

City Corvallis State Oregon Country U.S.A.

Telephone Zip/Postal Code 97330

Program Description (include equations) This lex file provides one keyword: ROWCOL$. Invocation:

ROWCOL$(<graphstring>)

The keyword accepts a single string argument of 0-8 characters. If argument is n characters (n<8) then characters n+1 through 8 default to nulls. Argument of >8 characters causes an "Invalid Arg" error.

Argument represents an 8 pixel by 8 pixel block of row- or column-oriented graphics. Result is an 8 pixel by 8 pixel block of column- or row-oriented graphics, respectively.

An argument or result of row-oriented graphics would actually be 8 bytes each containing 8 bits of column data from consecutive rows.

An argument or result of column-oriented graphics is actually 8 bytes each containing 8 bits of row data from consecutive columns. Here is a more hands-on explanation:

Necessary Accessories None

Supported Accessories N/A

Operating limits and warnings

File name(s) ROWCOL

Size of file(s) 119 bytes Additional RAM Requirement to run the program None

References

71-00002

2 of 7

+--+

CHAPTER 3 ROWCOL

+--+

<u>New Lex File</u> Indicates "RC:A" in VER$ string.

<u>Program Title:</u> String function for row/column graphics conversion.

<u>Category Number(s):</u> ???

<u>File Name(s):</u> ROWCOL.

<u>Primary Category Name:</u> ???

<u>Size of File(s):</u> 119 bytes.

<u>Additional RAM Requirement:</u> None.

<u>Abstract:</u> Lex file provides a keyword that allows easy conversion between row- and column-oriented graphics. Sample use: converting graphics data for HP82905B printer (column-oriented) into graphics data for Thinkjet printer (row-oriented).

<u>Necessary Accessories:</u> None.

<u>Supported Accessories:</u> N/A.

3.1 <u>Program Description</u>

This lex file provides one keyword: ROWCOL$. Invocation:

ROWCOL$(<graphstring>)

The keyword accepts a single string argument of 0-8 characters. If argument is n characters (n<8) then characters n+1 through 8 default to nulls. Argument of >8 characters causes an "Invalid Arg" error.

Argument represents an 8 pixel by 8 pixel block of row- or column-oriented graphics. Result is an 8 pixel by 8 pixel block of column- or row-oriented graphics, respectively.

An argument or result of row-oriented graphics would actually be 8 bytes each containing 8 bits of column data from consecutive rows.

3.2 Variable Definitions

N/A. 71-00002

3.3 Sample Usage

The following program converts a textfile containing graphics information for a THINKJET printer into graphics information for an HP82905B printer and prints that information.

The program is not fast; each line of print on the HP82905B takes about 45 seconds. But the use of the ROWCOL$ function on line 280 produces a drastic speed increase over what the program would take if it performed the equivalent manipulations in BASIC.

The program assumes that the file being dumped (called "MYFILE" here) contains THINKJET graphics directives of the form "<esc>*b<#bytes>W" (the preamble) followed by bytes of row graphics information. Any lines not of this format (as typically occur at the beginning and end of such files) are discarded without resulting in anything being printed.

71-00002 **SAMPLE PROBLEM SOLUTION**

DISPLAY CONTENTS	USER RESPONSE	COMMENTS

```
10 PRINT CHR$(27)&"&k2S"&CHR$(27)&"&l9d0L"
20 PWIDTH INF
30 DESTROY ALL
40 OPTION BASE 1
50 DIM C$[800],L(8),T$[8]
60 ASSIGN #1 TO MYFILE
70 ON ERROR GOTO 320
80 DESTROY R$ @ DIM R$(8)[100]
90 FOR I=1 TO 8
100 READ #1;R$(I)
110 IF R$(I)[1,3]#CHR$(27)&"*b" THEN 100
120 R$(I)=R$(I)[POS(R$(I),"W")+1]
130 NEXT I
140 OFF ERROR
150 GOSUB 170
160 GOTO 70
170 L9=0
180 FOR I=1 TO 8
190 L(I)=LEN(R$(I))
200 L9=MAX(L9,L(I))
210 NEXT I
220 C$=""
230 FOR I=1 TO L9
240 T$=""
250 FOR J=1 TO 8
260 T$=T$&R$(J)[I,I]&CHR$(0)[1+(I<=L(J))]
270 NEXT J
280 C$=C$&ROWCOL$(T$)
290 NEXT I
300 PRINT CHR$(27)&"*b"&STR$(LEN(C$))&"G";C$
310 RETURN
320 OFF ERROR
330 GOSUB 170
340 END
```

The following program converts a textfile containing graphics information for a THINKJET printer into graphics information for an HP 82905B printer and prints that information.

The program is not fast; each line of print on the HP 82905B takes about 45 seconds. But the use of the ROWCOL$ function on line 280 produces a drastic speed increase over what the program would take if it performed the equivalent manipulations in BASIC.

The program assumes that the file being dumped (called "MYFILE" here) contains THINKJET graphics directives of the form "<esc>*b<#bytes>W" (the preamble) followed by bytes of row graphics information. Any lines not of this format (as typically occur at the beginning and end of such files) are discarded without resulting in anything being printed.

Line 10 initializes the HP 82905B printer to compressed print mode, 9 lpi spacing and no-perforation-skip; appropriate settings for many graphics dumps. Lines 20-60 initialize the program. Line 70 traps the end-of-file condition. Line 80 reinitializes the row graphics string array to nulls. Lines 90-130 accumulate 8 rows of row graphics information for conversion to column graphics. Line 150 calls the subroutine to perform the actual conversion and line 160 loops back for more graphics information.

Lines 320-340 handle the printing of the final rows when the end-of-file is reached.

The conversion work is done in the subroutine at lines 170-310. Lines 170-210 build an array containing the lengths of the graphics data in each row, with L9 = the maximum length (note that line 120 stripped off the row graphics preamble from the line). Line 220 initializes the column graphics string to empty. Lines 230-290 build the column graphics string by grouping the row graphics bytes properly into T$ (line 260 insures a null space-filler if a row string is too short), converting T$ into column form with ROWCOL$ and appending it to the column graphics string (line 280). Line 300 prints the column graphics information prepended with the proper column graphics preamble ("<esc>*b<#bytes>G").

With slight modification, this program could print to a file instead of to a printer, producing a text file that can be easily and quickly printed on a column graphics printer.

71-00002

(Continuation Page)

Consider This pixel pattern:

```
                                         Row graphics pattern:
                  -------------------
                 | x   x x     x   |  4D
    (all numbers |   x x x   x x x |  EE
      in HEX)    | x x   x x     x |  9B
                 | x x x   x x x   |  77
                 |     x x x x x x |  FC
                 | x   x x x x   x |  BD
                 | x   x   x x x x |  F5
                 | x x x         x |  87
                  -------------------
   Column    :     E 8 F 3 7 7 5 F
   graphics  :     D E B 7 C A B 6
   pattern   :
```

The ROWCOL$ function will convert the byte sequence ED8EFB377C7A5BF6h into 4DEE9B77FCBDF587h and will also do the inverse (it is its own inverse).

This function provides a tool for BASIC to speak both row- and column-oriented graphics with minimal headache.

71-00002

Line 10 initializes the 82905B printer to compressed print mode, 9 lpi spacing and no-perforation-skip; appropriate settings for many graphics dumps. Lines 20-60 initialize the program. Line 70 traps the end-of-file condition. Line 80 reinitializes the row graphics string array to nulls. Lines 90-130 accumulate 8 rows of row graphics information for conversion to column graphics. Line 150 calls the subroutine to perform the actual conversion and line 160 loops back for more graphics information.

Lines 320-340 handle the printing of the final rows when the end-of-file is reached.

The conversion work is done in the subroutine at lines 170-310. Lines 170-210 build an array containing the lengths of the graphics data in each row, with L9 = the maximum length (note that line 120 stripped off the row graphics preamble from the line). Line 220 initializes the column graphics string to empty. Lines 230-290 build the column graphics string by grouping the row graphics bytes properly into T$ (line 260 insures a null space-filler if a row string is too short), converting T$ into column form with ROWCOL$ and appending it to the column graphics string (line 280). Line 300 prints the column graphics information prepended with the proper column graphics preamble ("<esc>*b<#bytes>G").

With slight modification, this program could print to a file instead of to a printer, producing a text file that can be easily and quickly printed on a column graphics printer.