DECUS
PROGRAM LIBRARY

DECUS NO.	8-732
TITLE	BAVIRF - A VIRTUAL FILE UDEF FOR OS/8 BASIC
AUTHOR	Stanley R. Vivian
COMPANY	University of Manitoba Faculty of Medicine Winnipeg, Canada
DATE	October 2, 1974
SOURCE LANGUAGE	PAL-8

GENERAL INFORMATION

Object Computer(s)	Any OS/8 system	Source Computer (if different)	
File Name	BAVIRF	Version No.	1
Title	BAVIRF - A Virtual File UDEF for OS/8 BASIC		
Author	Stanley R. Vivian		
Submitter (if other than author)			
Affiliation	Dept. of Pharmacology		
Address	University of Manitoba Faculty of Medicine		
	770 Bannatyne Ave., Winnipeg, Manitoba	Country	R3E 0W3 Canada
Monitor/Operating System	OS/8 V3 - OS/8 BASIC V3	DEC No.	
Core Storage Required	300_8	Starting Address	N/A
Peripherals Required	None		
Other Software Required	OS/8 BASIC V3	DEC or DECUS No.	
Source Language	PAL-8	Category	Numerical
Restrictions, Deficiencies, Problems			
Date of Planned or Possible Future Revisions	None		

TAPES AVAILABLE

Paper Tapes Object Binary ☐ Object ASCII ☐ Source ☒ Other ______

DECtape ☐ LINCtape ☐ Format ______ Magtape: 7 Track ☐ 9 Track ☐ BPI ______

Object Files ☐ Source Files ☐ Documentation Files ☐ Other ______

ABSTRACT

This overlay to OS/8 BASIC permits random access to the data in up to four numerical files - which may be of fixed or variable length. The maximum file length can contain 170,080 floating point numbers. The two functions, GET (F,L) and PUT(F,L,V), will retrieve, or deposit a value V, from or into location L of file F. Variable files are automatically expanded as needed. Users may switch from random to sequential access and vice versa. Full error checking is included to diagnose attempts to: access idle or non-numeric files; GET or PUT beyond the end of file; and, access data not within locations 1 to 170,080.

BAVIRF - A VIRTUAL FILE UDEF FOR OS/8 BASIC

DECUS Program Library Write-up DECUS NO. 8-732

DESCRIPTION

This program is an assembler language overlay to OS/8 BASIC. When interfaced to BASIC it will provide random access to data in numerical files. There may be up to four such files open and they may be of fixed or variable length. Mass storage permitting, each file may contain up to 170,080 floating point numbers. The two functions which provide this capability are defined in the BASIC source program with the UDEF command as follows:

```
UDEF GET(F,L),PUT(F,L,V)
```

where, F is the file number, 1 to 4, which must be open as a numeric fixed or variable file.

L is the location at which to retrieve or deposit the data. It must be a positive non-zero number within the range of the file and not greater than 170,080.

V is the floating point value to be deposited into the file.

Variable files will be automatically expanded as needed. Thus a variable file will contain the data from the largest location accessed after it is closed. After each random access the file pointer is set to the next sequential value in the file to facilitate switching from random to sequential access via the INPUT #F: or PRINT #F: commands. This may be necessary on occasion as the virtual file functions are not core resident with the extended numerical functions, such as LOG or EXP. To prevent excessive mass storage overlaying a GET function might be used to position the file pointer and data subsequent to it accessed sequentially with the use of extended numerical functions without overlaying.

A full error checking facility exists to diagnose the following indiscretions:

1. Addressing a location less than 1 or greater than 170,080 results in a fatal 'FM' or 'FO' error message.
2. Addressing an inactive file results in a fatal 'FI' error message.
3. Addressing a non-numeric file results in a fatal 'FN' error message.
4. Addressing a data point behond the largest file size, which is the largest empty file for variable files, results in a non-fatal 'RE' or 'WE' error message depending on whether the function was a GET or PUT. The end-of-file bit is also set.

The setting of the end-of-file bit after a 'RE' or 'WE' error message means that the IF END #F command will detect the end of file after a random access attempt to use a file behond its range.

The GET and PUT functions operate by first checking to see if the required block number for the requested data is core resident. If it is no mass storage input/output takes place and the data is directly accessed from core. If the requested data is not core resident a check is made to see if the currently resident data block for that file has been altered. If it has the block is written back to the mass storage device. The required block is then read into core and the buffer changed bit cleared.

EXAMPLE OF USE

The following program opens up a variable file, inputs N data points and writes them into the file. It then corrects any errors that might have been entered. Finally, the mean of the data is calculated and the file closed - making the data a permanent file.

```
10 REM --- VIRTUAL FILE EXAMPLE PROGRAM
20 REM
30 UDEF GET(F,L),PUT(F,L,V)
40 FILEVN #1:"DATA.BD"
50 PRINT "HOW MANY DATA POINTS";
60 INPUT N
70 PRINT "ENTER THE DATA"
80 FOR I=1 TO N
90 INPUT X
100 PRINT #1:X \ NEXT I
110 PRINT "HOW MANY ERRORS";
120 INPUT E
130 FOR J=1 TO E
140 PRINT "CHANGE LOCATION";
150 INPUT L
160 PRINT "FROM"; GET(1,L); "TO";
170 INPUT V
180 D=PUT(1,L,V)
190 NEXT J
200 S=0
210 FOR I=1 TO N
220 S=S+GET(1,I)
230 NEXT I
240 PRINT "THE MEAN IS"; S/N
250 CLOSE #1
260 STOP
270 END
```

In the above example all "reads" from the file had to be via the GET function as any attempt to INPUT from a variable file results in a fatal error.

```
HOW MANY DATA POINTS?17
ENTER THE DATA
?1 3 4 56 23 56 3 -2 4 78
?1 56 34 56 67 34 -23
HOW MANY ERRORS?3
CHANGE LOCATION?17
FROM-23 TO?-22
CHANGE LOCATION?2
FROM 3 TO?3.12
CHANGE LOCATION?6
FROM 56 TO?57
THE MEAN IS 26.6541

READY
BYE

.R PIP
*DATA.BD/L

 10/3/74
DATA  .BD    1 10/3/74
 541 FREE BLOCKS
```

INTERFACING WITH BASIC

Place the source paper tape of the overlay into the paper tape reader and give the following commands:

```
.R PIP
*BAVIRF.PA<PTR:(altmode)
.R PAL8
*BAVIRF.BN,<BAVIRF.PA
.R ABSLDR
*BAVIRF.BN(altmode)
.SAVE DSK BAVIRF
.R PIP
*SYS:BASIC.UF</D        [#]
*SYS:BASIC.UF<BAVIRF.SV/I(altmode)
.GET SYS BRTS
.ODT
1560/ 0240  3400
1561/ 0240  3403
7600G
.SAVE SYS BRTS
.GET SYS BRTS           [#]
.SAVE SYS BRTS          [#]
```

These commands ensure that the updated copies of BASIC.UF and BRTS.SV will be positioned in the same blocks on the system device. They may be considered optional.

MISCELLANEOUS

Version 1.0 of OS/8 BASIC

This overlay has not been tested with version 1.0 of OS/8 BASIC. It was however developed with the version 1.0 listing and hence should work with version 1.0. The overlay uses a number of routines from BRTS some of which have been relocated in version 3.0. Before attempting to interface the overlay with version 1.0 the following changes must be made to the source. Make three changes in the symbol definitions:

```
Make WE=3032
     WRBLK=3350
     BLINIT=3361
```

Inspiration

This overlay was inspired by the invaluable overlay to PS/8 FOCAL by Lawrence Moss, Cardiopulmonary Laboratory, University of Vermont College of Medicine, which provided random access to files from FOCAL.

```
/OS/8 BASIC VIRTUAL FILE UDEF

/       ********************************************
/       *                                          *
/       *  STANLEY R. VIVIAN                       *
/       *  DEPARTMENT OF PHARMACOLOGY              *
/       *  UNIVERSITY OF MANITOBA FACULTY OF MED.  *
/       *  770 BANNATYNE AVE                       *
/       *  WINNIPEG, MANITOBA                      *
/       *  CANADA                                  *
/       *  R3C 0W3                                 *
/       *                                          *
/       *  PHONE: (204) 786-3642                   *
/       *                                          *
/       ********************************************

/ADDS A VIRTUAL FILE CAPABILITY TO OS/8 BASIC.
/DATA CAN BE RETRIEVED FROM THE FILES AS FOLLOWS:
/
/       LET Y=GET(F,J) \ LET X=GET(4,J*100)

/DATA IS INSERTED INTO THE FILES AS FOLLOWS:
/
/       LET D=PUT(F,L,X(I)) \ LET D=PUT(4,J,V)
/               WHERE D IS A DUMMY VARIABLE


/ERROR MESSAGES:
/ 1.   AN ATTEMPT TO ADDRESS A LOCATION BEHOND 174,080, OR
/      LESS THAN 1, RESULTS IN AN 'FM' OR 'FO' ERROR
/      MESSAGE.
/ 2.   AN ATTEMPT TO ADDRESS AN INACTIVE FILE RESULTS
/      IN AN 'FI' ERROR.
/ 3.   AN ATTEMPT TO ADDRESS A NON-NUMERIC FILE, INCLUDING
/      FILE 0, RESULTS IN AN 'FN' ERROR.
/ 4.   AN ATTEMPT TO ADDRESS A BLOCK BEHOND THE LARGEST
/      FILE SIZE, (LARGEST EMPTY FILE FOR VARIABLE FILES),
/      RESULTS IN A NON-FATAL 'RE' OR 'WE' ERROR DEPENDING
/      ON WHETHER THE FUNCTION WAS A GET OR PUT.  THE EOF
/      BIT IS SET.

/THE SEQUENTIAL POINTER IS INITIALIZED TO POINT JUST
/BEHOND THE NUMBER JUST 'GOT' OR 'PUT' TO FACILITATE
/SWITCHING FROM RANDOM ACCESS TO SEQUENTIAL ACCESS.


/BRTS ENTRIES SHOULD BE INITIALIZED AS FOLLOWS:
/
/       .GET SYS BRTS
/       .ODT
/       1560/ 0240 3400
/       1561/ 0240 3403
/       7600G
/       .SAVE SYS BRTS
```

```
            /THE FOLLOWING SHOULD BE ADDED TO BASIC PROGRAMS:
            /
            /       20 UDEF GET(F,L),PUT(F,L,V)
            /
            /ANY OTHER UNIQUE FUNCTION NAMES MAY BE USED BUT
            /SHOULD BE IN THE ABOVE ORDER.


            /DEFINITIONS
      0037  FF=37
      0044  EXP=44
      0045  HORD=45
      0046  LORD=46
      0064  INSAV=64
      0113  ILOOPL=113
      0114  INTL=114
      0123  FIDLE=123
      0134  FGETL=134
      0135  FPUTL=135
      0136  FNORL=136
      0150  FTYPL=150
      0162  ENTNO=162
      0163  WORD0=163
      0164  WORD1=164
      0165  WORD2=165
      0166  WORD3=166
      0170  WORD5=170
      0171  WORD6=171
      0307  ARGPRE=307
      2000  SFN=2000
      2005  FN=2005
      3302  NEXREC=3302
      3010  RE=3010
      3025  WE=3025          /** FOR VERSION 1.0  WE=3032
      3352  WRBLK=3352       /                    WRBLK=3350
      3364  BLINIT=3364      /                    BLINIT=3361 **
      5400  FFSUB1=5400
      5600  FFMPY=5600
      5722  FFDIV=5722
      6117  FFSUB=6117
```

```
                /GET(F,L).  VIRTUAL FILE GET FUNCTION.
                /       F IS THE FILE NUMBER, 1 - 4
                /       L IS THE FILE LOCATION.
                /
                /PUT(F,L,V).  VIRTUAL FILE PUT FUNCTION.
                /       V IS THE VALUE TO BE INSERTED INTO THE FILE.

          3400  *3400

03400     0000  GET,    0               /GET ENTRY
03401     7001          IAC
03402     7410          SKP
03403     0000  PUT,    0               /PUT ENTRY
03404     7040          CMA
03405     3350          DCA RW          /GET=-2, PUT=-1
03406     4514          JMS I INTL
03407     3346          DCA FILENO      /FLAC HAS FILE #

03410     3037  LOCATE, DCA FF          /MODE 1
03411     4320          JMS ARG         /GET LOCATION
03412     4756          JMS I FSUBL     /BUFFERS START AT ZERO
03413     3566           F1             / NOT ONE - SUBTRACT 1
03414     4536          JMS I FNORL
03415     4535          JMS I FPUTL     /STORE AS A FLOAT
03416     3560           LOC

03417     1162  RESCHK, TAD ENTNO       /IS FILE CURRENT
03420     3347          DCA ORIGNO      /SAVE ORIGINAL IN CASE THIS
                                        / IS A 'PRINT'
03421     4753          JMS I TABLEL    /CHECK ON IT

03422     1563  EOFB,   TAD I WORDO     /TURN OFF EOF BIT
03423     7112          CLL RTR
03424     7106          CLL RTL
03425     3563          DCA I WORDO

03426     4523  ACTIVE, JMS I FIDLE     /CHECK IF FILE IS ACTIVE

03427     4550  NUMER,  JMS I FTYPL     /CHECK IF NUMERIC
03430     5233          JMP FINDBK      /OK - GO ON

03431     5632  ERROR,  JMP I .+1       /GIVE AN 'FN' FATAL ERROR
03432     2005          FN              /ILLEGAL FILE NO. ENTRY

03433     4534  FINDBK, JMS I FGETL     /CALCULATE BLOCK
03434     3560           LOC
03435     4754          JMS I FDIVL
03436     3563           F85
03437     4536          JMS I FNORL
03440     4514          JMS I INTL
03441     3351          DCA BLOCK       /INT((LOC-1)/85)

03442     1351  CORE,   TAD BLOCK       /IS BLOCK IN CORE
03443     1570          TAD I WORD5
03444     7041          CIA
```

```
03445  1565           TAD I WORD2
03446  7640           SZA CLA
03447  4752           JMS I MASSL      /NO - GO GET IT

03450  1351  FINDOS,  TAD BLOCK        /CALCULATE OFFSET
03451  4326           JMS FLOAT
03452  4755           JMS I FMPYL
03453  3563            F85
03454  4757           JMS I FSUB1L
03455  3560            LOC
03456  4536           JMS I FNORL
03457  4514           JMS I INTL       /OFFSET=(LOC-1)-85*INT(BLOCK)
03460  3360           DCA LOC

03461  1360  POINT,   TAD LOC          /POINTER=OFFSET*3+BUFFSTART
03462  7104           CLL RAL
03463  1360           TAD LOC          /MULT BY 3
03464  1564           TAD I WORD1      /ADD BUFFER STARTING POINT
03465  3360           DCA LOC          /NOW HAS ACTUAL ADDRESS

03466  7001  MODE2,   IAC              /MODE 2 NEEDED NEXT
03467  3037           DCA FF

03470  2350           ISZ RW           /READ OR WRITE?
03471  5305           JMP READ

03472  7001  WRITE,   IAC              /GET V IN FLAC
03473  4320           JMS ARG
03474  1360           TAD LOC          /DO THE 'PUT'
03475  6211           CDF 10
03476  4535           JMS I FPUTL
03477  7737  K7737,   7737             /USE THE SPACE

03500  1563  CHANGE,  TAD I WORD0      /SET THE BUFFER CHANGED BIT
03501  0277           AND K7737
03502  1310           TAD K40
03503  3563           DCA I WORD0
03504  5311           JMP EXIT

03505  1360  READ,    TAD LOC          /JUST GO GET IT
03506  6211           CDF 10
03507  4534           JMS I FGETL
03510  0040  K40,     40               /USE THE SPACE

03511  1360  EXIT,    TAD LOC          /SET FILE POINTER TO POINT
03512  1343           TAD K3           / JUST AFTER LAST ACCESS
03513  3566           DCA I WORD3
03514  4336           JMS TABLER       /RESET ORIGINAL TABLE
03515  2350           ISZ RW           /WHICH WAY HOME
03516  5603           JMP I PUT
03517  5600           JMP I GET

03520  0000  ARG,     0                /GETS NEXT ARG
03521  3064           DCA INSAV        /AC=0, FIRST
03522  4724           JMS I KARG       /AC=1, SECOND
```

```
03523  4534             JMS I FGETL
03524  0307   KARG,     ARGPRE
03525  5720             JMP I ARG

03526  0000   FLOAT,    0                     /FLOAT ROUTINE
03527  3046             DCA LORD
03530  3045             DCA HORD
03531  1345             TAD K27
03532  3044             DCA EXP
03533  4536             JMS I FNORL
03534  7200             CLA
03535  5726             JMP I FLOAT

03536  0000   TABLER,   0                     /RESETS ORIGINAL TABLE
03537  1347             TAD ORIGNO            / IF NECESSARY
03540  3346             DCA FILENO
03541  4753             JMS I TABLEL          /SEE IF IT NEEDS CHANGING
03542  5736             JMP I TABLER

03543  0003   K3,       3
03544  0004   K4,       4
03545  0027   K27,      27
03546  0000   FILENO,   0
03547  0000   ORIGNO,   0
03550  0000   RW,       0
03551  0000   BLOCK,    0
03552  3600   MASSL,    MASS
03553  3646   TABLEL,   TABLE
03554  5722   FDIVL,    FFDIV
03555  5600   FMPYL,    FFMPY
03556  6117   FSUBL,    FFSUB
03557  5400   FSUB1L,   FFSUB1
03560  0000   LOC,      0
03561  0000             0
03562  0000             0
03563  0007   F85,      7
03564  2520             2520
03565  0000             0
03566  0001   F1,       1
03567  2000             2000
03570  0000             0
```

```
        3600    *3600

03600   0000    MASS,   0                       /GETS CORRECT BLOCK INTO CORE

03601   4676    UPDATE, JMS I WRBLKL            /WRITE BACK BLOCK IF
03602   4677            JMS I BLINIL            / CHANGED AND CLEAR

03603   7040    RDBLK,  CMA                     /SET UP CURRENT BLOCK TO
03604   1704            TAD I BLOCKL            / BLOCK-1 FOR ISZ
03605   1570            TAD I WORD5
03606   3565            DCA I WORD2

03607   1113    GETBLK, TAD ILOOPL              /MODIFY 'EOF' RETURN
03610   3274            DCA TEMP2               / TO RETURN TO THIS
03611   7001            IAC                     / ROUTINE
03612   1231            TAD EOFR
03613   3113            DCA ILOOPL
03614   4673            JMS I NEXREL            /GET NEW BLOCK

03615   7240    EXPCHK, CLA CMA                 /IS THIS AN EXPANDING
03616   1571            TAD I WORD6             / VARIABLE FILE
03617   7041            CIA
03620   1704            TAD I BLOCKL
03621   7750            SPA SNA CLA
03622   5226            JMP OUT                 /NO - TAKE NO ACTION

03623   7001    EXPAND, IAC                     /EXPAND THE FILE
03624   1704            TAD I BLOCKL
03625   3571            DCA I WORD6

03626   1274    OUT,    TAD TEMP2               /RESTORE NORMAL RETURN
03627   3113            DCA ILOOPL
03630   5600            JMP I MASS

03631   3631    EOFR,   .
03632   1274    RWERR,  TAD TEMP2               /RESTORE NORMAL RETURN
03633   3113            DCA ILOOPL
03634   7040            CMA                     /FORCE A RE-READ AFTER RE
03635   3565            DCA I WORD2             / OR WE ERROR
03636   4707            JMS I TABRL             /RESET ORIGINAL TABLE
03637   4702            JMS I FLOATL            /RETURN WITH ZERO FLAG
03640   2703            ISZ I RWL               /WHICH MESSAGE?
03641   1301            TAD REDIF               /WAS A GET - GIVE 'RE'
03642   1300            TAD WEL                 /SET UP FOR 'WE' (OR 'RE')
03643   3245            DCA .+2                 /PUT THE JUMP IN LINE
03644   5645            JMP I .+1
03645   3645            .                       /THE 'RE' OR 'WE' MESSAGE IS
                                                / NON-FATAL - PROCESSING CONTINUES

03646   0000    TABLE,  0                       /IS FILE CURRENT
03647   1162            TAD ENTNO
03650   7041            CIA
03651   1706            TAD I FILENL
03652   7650            SNA CLA
03653   5646            JMP I TABLE             /YES - DO NOTHING
```

```
03654  1113  MOVE,    TAD ILOOPL       /NO - CHANGE IT
03655  3274           DCA TEMP2        /BORROW THE 'SFN' ROUTINE
03656  7001           IAC              / ITS NOT A JMS SO FIX RETURN
03657  1265           TAD SFNRET
03660  3113           DCA ILOOPL
03661  1044           TAD EXP          /SAVE EXP AS THIS ROUTINE
03662  3275           DCA TEXP         / WRECKS IT
03663  1706           TAD I FILENL     /ENTER WITH NEW FILE #
03664  5705           JMP I CSFNL
03665  3665  SFNRET,  .
03666  1275           TAD TEXP         /RESTORE EXP
03667  3044           DCA EXP
03670  1274           TAD TEMP2        /RESET NORMAL RETURN
03671  3113           DCA ILOOPL
03672  5646           JMP I TABLE

03673  3302  NEXREL,  NEXREC
03674  0000  TEMP2,   0
03675  0000  TEXP,    0
03676  3352  WRBLKL,  WRBLK
03677  3364  BLINIL,  BLINIT
03700  3025  WEL,     WE
03701  7763  REDIF,   RE-WE
03702  3526  FLOATL,  FLOAT
03703  3550  RWL,     RW
03704  3551  BLOCKL,  BLOCK
03705  2001  CSFNL,   SFN+1
03706  3546  FILENL,  FILENO
03707  3536  TABRL,   TABLER

       4000  *4000;   NOP              /MAKE THE OVERLAY NORMAL
04000  7000
       4200  *4200;   NOP              / SIZE
04200  7000
       4400  *4400;   NOP
04400  7000

             $$$$$$$$$$
```

```
/OS/8 BASIC VIRTUAL FILE UDEF                PAL8-V9B  10/02/74  PAGE 4

ACTIVE 3426       LORD   0046
ARG    3520       MASS   3600
ARGPRE 0307       MASSL  3552
BLINIL 3677       MODE2  3466
BLINIT 3364       MOVE   3654
BLOCK  3551       NEXREC 3302
BLOCKL 3704       NEXREL 3673
CHANGE 3500       NUMER  3427
CORE   3442       ORIGNO 3547
CSFNL  3705       OUT    3626
ENTNO  0162       POINT  3461
EOFB   3422       PUT    3403
EOFR   3631       RDBLK  3603
ERROR  3431       RE     3010
EXIT   3511       READ   3505
EXP    0044       REDIF  3701
EXPAND 3623       RESCHK 3417
EXPCHK 3615       RW     3550
FDIVL  3554       RWERR  3632
FF     0037       RWL    3703
FFDIV  5722       SFN    2000
FFMPY  5600       SFNRET 3665
FFSUB  6117       TABLE  3646
FFSUB1 5400       TABLEL 3553
FGETL  0134       TABLER 3536
FIDLE  0123       TABRL  3707
FILENL 3706       TEMP2  3674
FILENO 3546       TEXP   3675
FINDBK 3433       UPDATE 3601
FINDOS 3450       WE     3025
FLOAT  3526       WEL    3700
FLOATL 3702       WORD0  0163
FMPYL  3555       WORD1  0164
FN     2005       WORD2  0165
FNORL  0136       WORD3  0166
FPUTL  0135       WORD5  0170
FSUBL  3556       WORD6  0171
FSUB1L 3557       WRBLK  3352
FTYPL  0150       WRBLKL 3676
F1     3566       WRITE  3472
F85    3563
GET    3400
GETBLK 3607
HORD   0045
ILOOPL 0113
INSAV  0064
INTL   0114
KARG   3524
K27    3545
K3     3543
K4     3544
K40    3510
K7737  3477
LOC    3560
LOCATE 3410
```