```
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFFFFFFFFFF       000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFFFFFFFFFF       000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFFFFFFFFFF       000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFF               000         000 RRR   RRR       RRR   RRR             TTT        LLL
FFF               000         000 RRR   RRR       RRR   RRR             TTT        LLL
FFF               000         000 RRR   RRR       RRR   RRR             TTT        LLL
FFF               000         000 RRR     RRR     RRR     RRR           TTT        LLL
FFF               000         000 RRR     RRR     RRR     RRR           TTT        LLL
FFF               000         000 RRR     RRR     RRR     RRR           TTT        LLL
FFF                 000000000     RRR         RRR RRR         RRR       TTT        LLLLLLLLLLLLLLL
FFF                 000000000     RRR         RRR RRR         RRR       TTT        LLLLLLLLLLLLLLL
FFF                 000000000     RRR         RRR RRR         RRR       TTT        LLLLLLLLLLLLLLL
```

FILEID**FORBACKSP

```
FFFFFFFFFF    000000    RRRRRRRR    BBBBBBBB      AAAAAA      CCCCCCCC  KK      KK    SSSSSSSS  PPPPPPPP
FFFFFFFFFF    000000    RRRRRRRR    BBBBBBBB      AAAAAA      CCCCCCCC  KK      KK    SSSSSSSS  PPPPPPPP
FF          00      00  RR      RR  BB      BB  AA      AA  CC          KK      KK  SS          PP      PP
FF          00      00  RR      RR  BB      BB  AA      AA  CC          KK      KK  SS          PP      PP
FF          00      00  RR      RR  BB      BB  AA      AA  CC          KK    KK    SS          PP      PP
FF          00      00  RR      RR  BB      BB  AA      AA  CC          KK    KK    SS          PP      PP
FFFFFFFF    00      00  RRRRRRRR    BBBBBBBB    AA      AA  CC          KKKKKK        SSSSSS    PPPPPPPP
FFFFFFFF    00      00  RRRRRRRR    BBBBBBBB    AA      AA  CC          KKKKKK        SSSSSS    PPPPPPPP
FF          00      00  RR   RR     BB      BB  AAAAAAAAAA  CC          KK    KK            SS  PP
FF          00      00  RR   RR     BB      BB  AAAAAAAAAA  CC          KK    KK            SS  PP
FF          00      00  RR    RR    BB      BB  AA      AA  CC          KK      KK          SS  PP
FF          00      00  RR    RR    BB      BB  AA      AA  CC          KK      KK          SS  PP
FF            000000    RR      RR  BBBBBBBB    AA      AA    CCCCCCCC  KK      KK  SSSSSSSS  PP
FF            000000    RR      RR  BBBBBBBB    AA      AA    CCCCCCCC  KK      KK  SSSSSSSS  PP


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE FOR$BACKSPACE (%TITLE'FORTRAN BACKSPACE statement'
;     2          0002  0                 IDENT = '1-010'         ! File name: FORBACKSP.B32 Edit:SBL1010
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !*                                                                           *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                      *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    13          0013  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
;    16          0016  1 !*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY         *
;    17          0017  1 !*  TRANSFERRED.                                                              *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
;    21          0021  1 !*  CORPORATION.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:  FORTRAN Support Library
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains routine FOR$BACKSPACE (unit.rlu.v),
;    36          0036  1 !       which implements the FORTRAN BACKSPACE statement.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  User Mode - AST re-entrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:       Jonathan M. Taylor, CREATION DATE: 17-OCT-1977
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! [Previous edit history removed.  SBL 24-Sept-1982]
;    45          0045  1 ! 1-008 - Move BUILTIN ACTUALCOUNT into the routine.  The next version of the
;    46          0046  1 !         BLISS compiler will demand this.  JBS 20-Aug-1980
;    47          0047  1 ! 1-009 - Implement speedup for disk files.  SBL 3-Jun-1983
;    48          0048  1 ! 1-010 - Remove external literal declarations.  SBL 18-Jun-1983
;    49          0049  1 !--
;    50          0050  1
```

```
;    52          0051  1 !
;    53          0052  1 ! PROLOGUE FILE:
;    54          0053  1 !
;    55          0054  1
;    56          0055  1 REQUIRE 'RTLIN:FORPROLOG';                      FORTRAN declarations
;    57          0121  1
;    58          0122  1 !
;    59          0123  1 ! TABLE OF CONTENTS:
;    60          0124  1 !
;    61          0125  1
;    62          0126  1 FORWARD ROUTINE
;    63          0127  1     FOR$BACKSPACE;                          ! FORTRAN BACKSPACE statement
;    64          0128  1
;    65          0129  1 !
;    66          0130  1 ! OWN STORAGE:
;    67          0131  1 !
;    68          0132  1 !    NONE
;    69          0133  1 !
;    70          0134  1 ! EXTERNAL REFERENCES:
;    71          0135  1 !
;    72          0136  1
;    73          0137  1 EXTERNAL ROUTINE
;    74          0138  1     FOR$$SIGNAL_STO,                        ! convert error number and SIGNAL
;    75          0139  1     FOR$$CB_PUSH : JSB_CB_PUSH NOVALUE,     ! Create LUB/ISB/RAB if needed
;    76          0140  1     FOR$$CB_POP : JSB_CB_POP NOVALUE,       ! Return I/O system to previous state
;    77          0141  1     FOR$$IOSTAT_HND;                        ! Condition handler
;    78          0142  1
```

```
;    80          0143  1 GLOBAL ROUTINE FOR$BACKSPACE (%SBTTL'FORTRAN BACKSPACE statement'
;    81          0144  1         UNIT,                                   ! Logical unit
;    82          0145  1         ERR_EQL                                 ! Error code (optional)
;    83          0146  1     ) =
;    84          0147  1
;    85          0148  1 !++
;    86          0149  1 ! FUNCTIONAL DESCRIPTION:
;    87          0150  1 !
;    88          0151  1 !       Implements the FORTRAN BACKSPACE statement.
;    89          0152  1 !
;    90          0153  1 ! FORMAL PARAMETERS:
;    91          0154  1 !
;    92          0155  1 !       UNIT.rl.v               logical unit to perform backspace
;    93          0156  1 !       ERR_EQL.rl.v            If 0 or omitted, all errors are signalled.
;    94          0157  1 !                               If non-zero, errors unwind to the caller.
;    95          0158  1 !
;    96          0159  1 ! IMPLICIT INPUTS:
;    97          0160  1 !
;    98          0161  1 !       NONE
;    99          0162  1 !
;   100          0163  1 ! IMPLICIT OUTPUTS:
;   101          0164  1 !
;   102          0165  1 !       NONE
;   103          0166  1 !
;   104          0167  1 ! ROUTINE VALUE:
;   105          0168  1 !
;   106          0169  1 !       The returned value is always a correct IOSTAT small integer
;   107          0170  1 !       FORTRAN error number.
;   108          0171  1 !
;   109          0172  1 ! SIDE EFFECTS:
;   110          0173  1 !
;   111          0174  1 !       SIGNAL_STOPs FOR$_BACERR if RMS REWIND error or RMS $GET error.
;   112          0175  1 !
;   113          0176  1 !--
;   114          0177  1
;   115          0178  2     BEGIN
;   116          0179  2
;   117          0180  2     GLOBAL REGISTER
;   118          0181  2         CCB = K_CCB_REG : REF $FOR$CCB_DECL;
;   119          0182  2
;   120          0183  2     BUILTIN
;   121          0184  2         ACTUALCOUNT;
;   122          0185  2
;   123          0186  2     LOCAL
;   124          0187  2         FOUND,                                  ! 1 if we have got the record
;   125          0188  2         L_UNWIND_ACTION : VOLATILE,             ! UNWIND action code
;   126          0189  2         L_ERR_EQL_PRES : VOLATILE;              ! 1 if ERR= present, 0 otherwise
;   127          0190  2
;   128          0191  2     ENABLE
;   129          0192  2         FOR$$IOSTAT_HND (L_UNWIND_ACTION, L_ERR_EQL_PRES);      ! LUB cleanup with ERR= and IOSTAT
;   130          0193  2
;   131          0194  2     !+
;   132          0195  2     ! Determine if ERR= is present.
;   133          0196  2     !-
;   134          0197  2
;   135          0198  2     IF ACTUALCOUNT () GTR 1 THEN L_ERR_EQL_PRES = .ERR_EQL ELSE L_ERR_EQL_PRES = 0;
;   136          0199  2
```

```
;  137          0200  2      !+
;  138          0201  2      ! Unwind action is NO-OP (no LUB to pop).
;  139          0202  2      !-
;  140          0203  2
;  141          0204  2      L_UNWIND_ACTION = FOR$K_UNWINDNOP;
;  142          0205  2
;  143          0206  2      !-
;  144          0207  2      ! Get a LUB for this logical unit.
;  145          0208  2      ! On return, CCB points to the current control block.
;  146          0209  2      !-
;  147          0210  2
;  148          0211  2      FOR$$CB_PUSH (.UNIT, LUB$K_LUN_MIN);
;  149          0212  2
;  150          0213  2      !+
;  151          0214  2      ! Unwind action (in case error occurs) is to pop the LUB/RAB/ISB.
;  152          0215  2      !-
;  153          0216  2
;  154          0217  2      L_UNWIND_ACTION = FOR$K_UNWINDPOP;
;  155          0218  2
;  156          0219  2      !+
;  157          0220  2      ! Check the LUB.  If the file is not open, then this is a no-op.
;  158          0221  2      !-
;  159          0222  2
;  160          0223  2      IF .CCB [LUB$V_OPENED]
;  161          0224  2      THEN
;  162          0225  3          BEGIN
;  163          0226  3
;  164          0227  3          BIND
;  165          0228  3              FAB = CCB: REF $FOR$FAB_CCB_STRUCT,
;  166          0229  3              NAM = CCB: REF $FOR$NAM_CCB_STRUCT,
;  167          0230  3              FAB_DEV = FAB [FAB$L_DEV]: BLOCK [4, BYTE];
;  168          0231  3
;  169          0232  3          !+
;  170          0233  3          ! Ensure that the file is sequential organization and access.
;  171          0234  3          !-
;  172          0235  3
;  173          0236  3          IF .CCB [LUB$V_NOTSEQORG] OR
;  174          0237  3             .CCB [LUB$V_DIRECT] OR
;  175          0238  3             .CCB [LUB$V_APPEND]
;  176          0239  3          THEN
;  177          0240  3              RETURN FOR$$SIGNAL_STO (FOR$K_BACERR,0,
;  178          0241  3                  FOR$_OPEREQSEQ); ! Operation requires sequential org. and access
;  179          0242  3
;  180          0243  3          !+
;  181          0244  3          ! Ensure that this is a tape or disk file and is not a PPF.
;  182          0245  3          !-
;  183          0246  3
;  184          0247  3          IF NOT (.FAB_DEV [DEV$V_SQD] OR .FAB_DEV [DEV$V_RND]) OR
;  185          0248  3             .NAM [NAM$V_PPF]
;  186          0249  3          THEN
;  187          0250  3              RETURN FOR$$SIGNAL_STO (FOR$K_BACERR,0,
;  188          0251  3                  FOR$_OPEREQDIS); ! Operation requires disk or tape file
;  189          0252  3
;  190          0253  3          !+
;  191          0254  3          ! Indicate that we have not yet positioned to the record.
;  192          0255  3          !-
;  193          0256  3
```

```
;  194          0257  3          FOUND = 0;
;  195          0258  3
;  196          0259  3          !+
;  197          0260  3          ! If we have read at least two records in the file, try to use a
;  198          0261  3          ! fast method of finding the record.  However, we can't do the
;  199          0262  3          ! fast method if FAB$V_SQO is set (set for DECnet files).
;  200          0263  3          !-
;  201          0264  3
;  202          0265  3          IF .CCB [LUB$L_LOG_RECNO] GTRU 2 AND NOT .FAB [FAB$V_SQO]
;  203          0266  3          THEN
;  204          0267  4              BEGIN
;  205          0268  4              LOCAL
;  206          0269  4                  DEST_RECNO;     ! Destination record number.
;  207          0270  4
;  208          0271  4              !+
;  209          0272  4              ! Compute the number of the record before the current one.  Note
;  210          0273  4              ! that since LUB$L_LOG_RECNO contains the number of the NEXT record,
;  211          0274  4              ! we must subtract 2.
;  212          0275  4              !-
;  213          0276  4
;  214          0277  4              DEST_RECNO = .CCB [LUB$L_LOG_RECNO] - 2;
;  215          0278  4
;  216          0279  4              !+
;  217          0280  4              ! Use direct-access positioning if fixed-length records.
;  218          0281  4              !-
;  219          0282  4
;  220          0283  4              IF .CCB [LUB$V_FIXED]
;  221          0284  4              THEN
;  222          0285  5                  BEGIN
;  223          0286  5
;  224          0287  5                  !+
;  225          0288  5                  ! Set record number to previous record and do a $GET.
;  226          0289  5                  ! If it succeeds, then set FOUND to indicate we have
;  227          0290  5                  ! the record we want.
;  228          0291  5                  !-
;  229          0292  5
;  230          0293  5                  CCB [RAB$L_KBF] = DEST_RECNO;
;  231          0294  5                  CCB [RAB$B_RAC] = RAB$C_KEY;
;  232          0295  6                  IF $GET (RAB = .CCB)
;  233          0296  5                  THEN
;  234          0297  6                      BEGIN
;  235          0298  6                      CCB [LUB$L_LOG_RECNO] = .CCB [LUB$L_LOG_RECNO] - 1;
;  236          0299  6                      FOUND = .FOUND + 1;
;  237          0300  5                      END;
;  238          0301  4                  END;
;  239          0302  4
;  240          0303  4          !+
;  241          0304  4          ! If RFA cacheing is in effect, look in the cache to see if we
;  242          0305  4          ! have the desired record in the cache.  If so, do an RFA $GET of
;  243          0306  4          ! it.  Note that RFA cacheing is not enabled for fixed-length
;  244          0307  4          ! records, so we don't have the record yet.
;  245          0308  4          !-
;  246          0309  4
;  247          0310  4          IF .CCB [LUB$V_RFA_CACHE_ENABLE]
;  248          0311  4          THEN
;  249          0312  5              BEGIN
;  250          0313  5              LOCAL
```

```
;  251          0314  5                    RCE: REF RCE_R_RCE_STRUCT;  ! RFA Cache Entry
;  252          0315  5
;  253          0316  5                RCE = .CCB [LUB$A_RFA_CACHE_PTR];       ! Get current entry
;  254          0317  5
;  255          0318  5                !+
;  256          0319  5                ! Look through cache for the right record number.
;  257          0320  5                !-
;  258          0321  5
;  259          0322  5                DECRU I FROM RCE_K_CACHE_SIZE TO 1 DO
;  260          0323  6                    BEGIN
;  261          0324  6                    IF .RCE [RCE_L_LOG_RECNO] EQLU .DEST_RECNO
;  262          0325  6                    THEN
;  263          0326  7                        BEGIN
;  264          0327  7                        CCB [RAB$L_RFA0] = .RCE [RCE_L_RFA0];
;  265          0328  7                        CCB [RAB$W_RFA4] = .RCE [RCE_W_RFA4];
;  266          0329  7                        CCB [RAB$B_RAC] = RAB$C_RFA;
;  267          0330  8                        IF $GET (RAB = .CCB)
;  268          0331  7                        THEN
;  269          0332  8                            BEGIN
;  270          0333  8                            CCB [LUB$A_RFA_CACHE_PTR] = .RCE [RCE_A_NEXT];
;  271          0334  8                            CCB [LUB$L_LOG_RECNO] = .CCB [LUB$L_LOG_RECNO] - 1;
;  272          0335  8                            FOUND = .FOUND + 1;
;  273          0336  7                            END;
;  274          0337  7                        EXITLOOP;
;  275          0338  6                        END;
;  276          0339  6                    RCE = .RCE [RCE_A_PREV];    ! Get next entry
;  277          0340  5                    END;
;  278          0341  4                END;
;  279          0342  3            END;
;  280          0343  3
;  281          0344  3            !+
;  282          0345  3            ! If we haven't found the record yet, rewind and forward space.
;  283          0346  3            !-
;  284          0347  3
;  285          0348  3            IF NOT .FOUND
;  286          0349  3            THEN
;  287          0350  4                BEGIN
;  288          0351  4
;  289          0352  4                LOCAL
;  290          0353  4                    RCE: REF RCE_R_RCE_STRUCT;      ! RFA Cache entry
;  291          0354  4
;  292          0355  4                RCE = .CCB [LUB$A_RFA_CACHE_PTR];
;  293          0356  4                CCB [RAB$B_RAC] = RAB$C_SEQ;
;  294          0357  4
;  295          0358  4                !+
;  296          0359  4                ! If RMS can't rewind the file then fail.
;  297          0360  4                !-
;  298          0361  4
;  299          0362  4                IF NOT $REWIND (RAB = .CCB) THEN
;  300          0363  4                    RETURN FOR$$SIGNAL_STO (FOR$K_BACERR);
;  301          0364  4
;  302          0365  4                !+
;  303          0366  4                ! Now read records until we're positioned one record before
;  304          0367  4                ! the last record read.
;  305          0368  4                !-
;  306          0369  4
;  307          0370  4                IF .CCB [LUB$L_LOG_RECNO] GTRU 1    ! i.e. we're sitting somewhere in the file
```

```
;  308     0371  ;                THEN
;  309     0372  5                    BEGIN
;  310     0373  5
;  311     0374  5                    LOCAL
;  312     0375  5                        I;                              ! local temp for counting
;  313     0376  5
;  314     0377  5                    I = .CCB [LUB$L_LOG_RECNO] - 1; ! Point to previous record
;  315     0378  5                    CCB [LUB$L_LOG_RECNO] = 1;
;  316     0379  5
;  317     0380  5                    WHILE .CCB [LUB$L_LOG_RECNO] LSSU .I DO
;  318     0381  6                        BEGIN
;  319     0382  6
;  320     0383  6                        !+
;  321     0384  6                        ! Call RMS to get next record.
;  322     0385  6                        !-
;  323     0386  6
;  324     0387  7                        IF NOT $GET (RAB = .CCB)
;  325     0388  6                        THEN
;  326     0389  6                            RETURN FOR$$SIGNAL_STO (FOR$K_BACERR);
;  327     0390  6
;  328     0391  6                        !+
;  329     0392  6                        ! Make entry in RFA cache if enabled.
;  330     0393  6                        !-
;  331     0394  6
;  332     0395  6                        IF .CCB [LUB$V_RFA_CACHE_ENABLE]
;  333     0396  6                        THEN
;  334     0397  7                            BEGIN
;  335     0398  7                            RCE [RCE_L_LOG_RECNO] = .CCB [LUB$L_LOG_RECNO];
;  336     0399  7                            $LIB$MOVQ (CCB [RAB$W_RFA], RCE [RCE_Q_RFA]);
;  337     0400  7                            RCE = .RCE [RCE_A_NEXT];
;  338     0401  6                            END;
;  339     0402  6
;  340     0403  6                        !+
;  341     0404  6                        ! If segmented record control, check the validity
;  342     0405  6                        ! of the records and for end-of-file.  Read the
;  343     0406  6                        ! following segments of the segmented record until
;  344     0407  6                        ! last record control info is seen.
;  345     0408  6                        !-
;  346     0409  6
;  347     0410  6                        IF .CCB [LUB$V_SEGMENTED]
;  348     0411  6                        THEN
;  349     0412  7                            BEGIN
;  350     0413  7                            UNTIL
;  351     0414  8                                BEGIN
;  352     0415 10                                IF ((.CCB [RAB$W_RSZ] EQL 0)
;  353     0416 10                                    OR ((.CCB [RAB$W_RSZ] GEQU 2) AND
;  354     0417  9                                     (.(.CCB [RAB$L_RBF])<2, 14> NEQ 0)))
;  355     0418  8                                THEN
;  356     0419  8                                    RETURN FOR$$SIGNAL_STO (FOR$K_SEGRECFOR);
;  357     0420  8                                .(.CCB [RAB$L_RBF])<1, 1>
;  358     0421  8                                END
;  359     0422  7                            DO
;  360     0423  8                                BEGIN
;  361     0424  9                                IF NOT $GET (RAB = .CCB)
;  362     0425  8                                THEN
;  363     0426  8                                    RETURN FOR$$SIGNAL_STO (FOR$K_BACERR);
;  364     0427  7                                END;
```

```
;  365     0428  6                          END;
;  366     0429  6
;  367     0430  6                      CCB [LUB$L_LOG_RECNO] = .CCB [LUB$L_LOG_RECNO] + 1;
;  368     0431  5                      END;
;  369     0432  4                  END;
;  370     0433  4              !+
;  371     0434  4              ! Record found - set RFA cache entry pointer.
;  372     0435  4              !-
;  373     0436  4
;  374     0437  4              CCB [LUB$A_RFA_CACHE_PTR] = .RCE;
;  375     0438  3              END;
;  376     0439  3
;  377     0440  2          END;
;  378     0441  2
;  379     0442  2      !+
;  380     0443  2      ! Return I/O system to previous state.
;  381     0444  2      !-
;  382     0445  2
;  383     0446  2      FOR$$CB_POP ();
;  384     0447  2      RETURN 0;                                   ! Success IOSTAT value
;  385     0448  1      END;


                                .TITLE  FOR$BACKSPACE FORTRAN BACKSPACE statement
                                .IDENT  \1-010\

                                .EXTRN  FOR$$SIGNAL_STO
                                .EXTRN  FOR$$CB_PUSH, FOR$$CB_POP
                                .EXTRN  FOR$$IOSTAT_HND
                                .EXTRN  SYS$GET, SYS$REWIND

                                .PSECT  _FOR$CODE,NOWRT,  SHR,  PIC,2

                        09FC 00000      .ENTRY  FOR$BACKSPACE, Save R2,R3,R4,R5,R6,R7,R8,-  ; 0143
                                                R11                                        ;
                58 00000000G    00  9E 00002      MOVAB   FOR$$SIGNAL_STO, R8               ;
                57 00000000G    00  9E 00009      MOVAB   SYS$GET, R7                       ;
                5E              0C  C2 00010      SUBL2   #12, SP                           ;
                      04        AE  7C 00013      CLRQ    L_ERR_EQL_PRES                    ; 0178
                6D      0146    CF  DE 00016      MOVAL   24$, (FP)                         ;
                01              6C  91 0001B      CMPB    (AP), #1                          ; 0198
                                07  1B 0001E      BLEQU   1$                                ;
          04    AE        08    AC  D0 00020      MOVL    ERR_EQL, L_ERR_EQL_PRES           ;
                                03  11 00025      BRB     2$                                ;
                                04  AE  D4 00027 1$:     CLRL    L_ERR_EQL_PRES            ;
          08    AE                  01  D0 0002A 2$:     MOVL    #1, L_UNWIND_ACTION       ; 0204
                                    50  D4 0002E        CLRL    R0                        ; 0211
                52              04  AC  D0 00030      MOVL    UNIT, R2                  ;
                    000000000G  00  16 00034      JSB     FOR$$CB_PUSH              ;
                                08  AE  D4 0003A      CLRL    L_UNWIND_ACTION           ; 0217
                56              FC  AB  9E 0003D      MOVAB   -4(CCB), R6               ; 0223
                03                  66  E8 00041      BLBS    (R6), 3$                  ;
                                0110  31 00044      BRW     23$                       ;
                52      0084    CB  9E 00047 3$:     MOVAB   132(R11), R2              ; 0230
    08    A1    AB              03  E0 0004C      BBS     #3, -95(CCB), 4$          ; 0236
    04          66              04  E0 00051      BBS     #4, (R6), 4$              ; 0237
    08          66              0D  E1 00055      BBC     #13, (R6), 5$             ; 0238
```

```
                                    0018884C  8F  DD 00059 4$:   PUSHL   #1607756                          ; 0240
                                              13  11 0005F       BRB     8$
                              04        62      05  E0 00061 5$:   BBS     #5, (R2), 6$                   ; 0247
                              05        62      1C  E1 00065       BBC     #28, (R2), 7$
                                        0D  00CA  CB  E9 00069 6$:   BLBC    202(CCB), 9$                   ; 0248
                                    00188844  8F  DD 0006E 7$:   PUSHL   #1607748                         ; 0250
                                        7E      17  7D 00074 8$:   MOVQ    #23, -(SP)
                                        68      03  FB 00077       CALLS   #3, FOR$$SIGNAL_STO
                                                04  0007A       RET
                                                55  D4 0007B 9$:   CLRL    FOUND                          ; 0257
                                    53      E0  AB  9E 0007D       MOVAB   -32(CCB), R3                   ; 0265
                                    02      63  D1 00081       CMPL    (R3), #2
                                                5B  1B 00084       BLEQU   13$
                              56        48  AB  06  E0 00086       BBS     #6, 72(CCB), 13$
                              6E            63  02  C3 0008B       SUBL3   #2, (R3), DEST_RECNO           ; 0277
                              14            66  0A  E1 0008F       BBC     #10, (R6), 10$                 ; 0283
                                    30      AB  6E  9E 00093       MOVAB   DEST_RECNO, 48(CCB)            ; 0293
                                    1E      AB  01  90 00097       MOVB    #1, 30(CCB)                    ; 0294
                                                5B  DD 0009B       PUSHL   CCB                            ; 0295
                                        67      01  FB 0009D       CALLS   #1, SYS$GET
                                        04      50  E9 000A0       BLBC    R0, 10$
                                                63  D7 000A3       DECL    (R3)                           ; 0298
                                                55  D6 000A5       INCL    FOUND                          ; 0299
                              35        A1  AB  05  E1 000A7 10$:   BBC     #5, -95(CCB), 13$              ; 0310
                                        52  CC  AB  D0 000AC       MOVL    -52(CCB), RCE                  ; 0316
                                        54      14  D0 000B0       MOVL    #20, I                         ; 0324
                                        6E  08  A2  D1 000B3 11$:   CMPL    8(RCE), DEST_RECNO
                                                20  12 000B7       BNEQ    12$
                                    10  AB  0C  A2  D0 000B9       MOVL    12(RCE), 16(CCB)               ; 0327
                                    14  AB  10  A2  B0 000BE       MOVW    16(RCE), 20(CCB)               ; 0328
                                    1E  AB      02  90 000C3       MOVB    #2, 30(CCB)                    ; 0329
                                                5B  DD 000C7       PUSHL   CCB                            ; 0330
                                        67      01  FB 000C9       CALLS   #1, SYS$GET
                                        12      50  E9 000CC       BLBC    R0, 13$
                                    CC  AB      62  D0 000CF       MOVL    (RCE), -52(CCB)                ; 0333
                                                63  D7 000D3       DECL    (R3)                           ; 0334
                                                55  D6 000D5       INCL    FOUND                          ; 0335
                                                08  11 000D7       BRB     13$                            ; 0326
                                        52      04  A2  D0 000D9 12$:   MOVL    4(RCE), RCE               ; 0339
                                                54  D7 000DD       DECL    I                              ; 0322
                                                D2  12 000DF       BNEQ    11$
                                        73      55  E8 000E1 13$:   BLBS    FOUND, 23$                     ; 0348
                                        54  CC  AB  D0 000E4       MOVL    -52(CCB), RCE                  ; 0355
                                            1E  AB  94 000E8       CLRB    30(CCB)                        ; 0356
                                                5B  DD 000EB       PUSHL   CCB                            ; 0362
                          00000000G     00      01  FB 000ED       CALLS   #1, SYS$REWIND
                                        52      50  E9 000F4       BLBC    R0, 19$
                                        01      63  D1 000F7       CMPL    (R3), #1                       ; 0370
                                                57  1B 000FA       BLEQU   22$
                          52                63  01  C3 000FC       SUBL3   #1, (R3), I                    ; 0377
                                            63  01  D0 00100       MOVL    #1, (R3)                       ; 0378
                                        52      63  D1 00103 14$:   CMPL    (R3), I                        ; 0380
                                                4B  1E 00106       BGEQU   22$
                                                5B  DD 00108       PUSHL   CCB                            ; 0387
                                        67      01  FB 0010A       CALLS   #1, SYS$GET
                                        39      50  E9 0010D       BLBC    R0, 19$
                          0C        A1  AB      05  E1 00110       BBC     #5, -95(CCB), 15$              ; 0395
```

```
                        08   A4                63  DO 00115          MOVL    (R3), 8(RCE)                          ; 0398
                        0C   A4          10     AB  7D 00119          MOVQ    16(CCB), 12(RCE)                      ; 0399
                             54                 64  DO 0011E          MOVL    (RCE), RCE                            ; 0400
                  2A         66                 0B  E1 00121 15$:     BBC     #11, (R6), 21$                        ; 0410
                                          22    AB  B5 00125 16$:     TSTW    34(CCB)                               ; 0415
                                                0E  13 00128          BEQL    17$                                   ;
                             02           22    AB  B1 0012A          CMPW    34(CCB), #2                           ; 0416
                                                0C  1F 0012E          BLSSU   18$                                   ;
                        FFFC 8F           28    BB  B3 00130          BITW    @40(CCB), #65532                      ; 0417
                                                04  13 00136          BEQL    18$                                   ;
                                                23  DD 00138 17$:     PUSHL   #35                                   ; 0419
                                                0F  11 0013A          BRB     20$                                   ;
                  0E        28   BB                01  E0 0013C 18$:     BBS     #1, @40(CCB), 21$                     ; 0420
                                                5B  DD 00141          PUSHL   CCB                                   ; 0424
                             67                 01  FB 00143          CALLS   #1, SYS$GET                           ;
                             DC                 50  E8 00146          BLBS    R0, 16$                               ;
                                                17  DD 00149 19$:     PUSHL   #23                                   ; 0426
                             68                 01  FB 0014B 20$:     CALLS   #1, FOR$$SIGNAL_STO                   ;
                                                04  0014E            RET                                           ;
                                                63  D6 0014F 21$:     INCL    (R3)                                  ; 0430
                                                B0  11 00151          BRB     14$                                   ; 0380
                        CC   AB                 54  DO 00153 22$:     MOVL    RCE, -52(CCB)                         ; 0437
                                00000000G       00  16 00157 23$:     JSB     FOR$$CB_POP                           ; 0446
                                                50  D4 0015D          CLRL    R0                                    ; 0447
                                                04  0015F            RET                                           ; 0448
                                              0000  00160 24$:     .WORD   Save nothing                          ; 0178
                             50           08    AC  DO 00162          MOVL    8(AP), R0                             ;
                             50           04    A0  DO 00166          MOVL    4(R0), R0                             ;
                                          F8    A0  9F 0016A          PUSHAB  L_ERR_EQL_PRES                        ;
                                          FC    A0  9F 0016D          PUSHAB  L_UNWIND_ACTION                       ;
                                                02  DD 00170          PUSHL   #2                                    ;
                                                5E  DD 00172          PUSHL   SP                                    ;
                                    7E    04    AC  7D 00174          MOVQ    4(AP), -(SP)                          ;
                          00000000G 00          03  FB 00178          CALLS   #3, FOR$$IOSTAT_HND                   ;
                                                04  0017F            RET                                           ;

; Routine Size:  384 bytes,    Routine Base:  _FOR$CODE + 0000

;  386          0449  1
;  387          0450  1 END
;  388          0451  1
;  389          0452  0 ELUDOM



;                                    PSECT SUMMARY
;
;      Name                          Bytes                          Attributes
;
;  _FOR$CODE                          384  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                              Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776        20         0         581        00:01.0
;  _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1    711       201        28          52        00:00.5
;  _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1     36         1         2           8        00:00.1
```

```
;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FORBACKSP/OBJ=OBJ$:FORBACKSP MSRC$:FO BACKSP/UPDATE=(ENH$:FORBACKSP
;       )

; Size:         384 code + 0 data bytes
; Run Time:       00:11.8
; Elapsed Time:   00:37.1
; Lines/CPU Min:   2292
; Lexemes/CPU-Min: 15580
; Memory Used:  187 pages
; Compilation Complete
```

COMRS0WD
LIS

FORDATEDS
LIS

FORDECOMO
LIS

FORCLB
LIS

COMSETST
LIS

FORASSOC
LIS

FORCLOSEF
LIS

FORDATE
LIS

FORCLOSE
LIS

FORDECOMF
LIS

FORDELETE
LIS

COMRAD50
LIS

COMUSEREX
LIS

FORBITOPS
LIS

FORDEFINE
LIS

FORBACKSP
LIS

FORDISPA
LIS

FORCUTRT
LIS