```
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS  FFFFFFFFFF  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  HH      HH    AAAAAA
BBBBBBBB      AAAAAA      SSSSSSSS  FFFFFFFFFF  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  HH      HH    AAAAAA
BB      BB   AA      AA  SS        FF          EE              TT       CC            HH      HH   AA      AA
BB      BB   AA      AA  SS        FF          EE              TT       CC            HH      HH   AA      AA
BB      BB   AA      AA  SS        FF          EE              TT       CC            HH      HH   AA      AA
BB      BB   AA      AA  SS        FF          EE              TT       CC            HH      HH   AA      AA
BBBBBBBB     AA      AA    SSSSSS  FFFFFFFF    EEEEEEEE        TT       CC            HHHHHHHHHH   AA      AA
BBBBBBBB     AA      AA    SSSSSS  FFFFFFFF    EEEEEEEE        TT       CC            HHHHHHHHHH   AA      AA
BB      BB   AAAAAAAAAA          SS  FF          EE              TT       CC            HH      HH   AAAAAAAAAA
BB      BB   AAAAAAAAAA          SS  FF          EE              TT       CC            HH      HH   AAAAAAAAAA
BB      BB   AA      AA          SS  FF          EE              TT       CC            HH      HH   AA      AA    ....
BB      BB   AA      AA          SS  FF          EE              TT       CC            HH      HH   AA      AA    ....
BBBBBBBB     AA      AA  SSSSSSSS    FF          EEEEEEEEEE      TT         CCCCCCCC    HH      HH   AA      AA    ....
BBBBBBBB     AA      AA  SSSSSSSS    FF          EEEEEEEEEE      TT         CCCCCCCC    HH      HH   AA      AA    ....

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 BAS$FETCH_ADDR (                      ! Fetch address of array element
;     2          0002  0                 IDENT = '1-004'              ! File: BASFETCHA.B32 Edit: PLL1004
;     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:  BASIC Language Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module calculates the address of a non-virtual array
;    36          0036  1 !       element.  It is called by the compiled code for the LOC
;    37          0037  1 !       function and for arrays passed as parameters.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:  VAX-11 User Mode
;    40          0040  1 !
;    41          0041  1 ! AUTHOR: Pamela L. Levesque, CREATION DATE: 19-FEB-1982
;    42          0042  1 !
;    43          0043  1 ! MODIFIED BY:
;    44          0044  1 !
;    45          0045  1 ! 1-001 - Original.  PLL 19-Feb-1982
;    46          0046  1 ! 1-002 - Add support for decimal arrays.  This involves calculating the
;    47          0047  1 !         size of elements in bytes (the length in the descriptor is the
;    48          0048  1 !         number of digits not including the sign), and using that length
;    49          0049  1 !         to calculate the linear index.  PLL 12-Mar-1982
;    50          0050  1 ! 1-003 - Offset for 1st index is 1, not 2.  PLL 19-Mar-1982
;    51          0051  1 ! 1-004 - Return address of descriptor for dynamic strings.  PLL 29-Mar-1982
;    52          0052  1 !--
;    53          0053  1
;    54          0054  1 !<BLF/PAGE>
```

```
;    56          0055  1 !
;    57          0056  1 ! SWITCHES:
;    58          0057  1 !
;    59          0058  1
;    60          0059  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    61          0060  1
;    62          0061  1 !
;    63          0062  1 ! LINKAGES:
;    64          0063  1 !
;    65          0064  1 !    NONE
;    66          0065  1 !
;    67          0066  1 !
;    68          0067  1 ! TABLE OF CONTENTS:
;    69          0068  1 !
;    70          0069  1
;    71          0070  1 FORWARD ROUTINE
;    72          0071  1     BAS$FETCH_ADDR;                         ! Fetch address of array element
;    73          0072  1
;    74          0073  1 !
;    75          0074  1 ! INCLUDE FILES:
;    76          0075  1 !
;    77          0076  1
;    78          0077  1 REQUIRE 'RTLIN:RTLPSECT';                   ! Macros for defining psects
;    79          0172  1
;    80          0173  1 LIBRARY 'RTLSTARLE';                        ! System symbols
;    81          0174  1
;    82          0175  1 !
;    83          0176  1 ! MACROS:
;    84          0177  1 !
;    85          0178  1 !       NONE
;    86          0179  1 !
;    87          0180  1 ! EQUATED SYMBOLS:
;    88          0181  1 !
;    89          0182  1 !       NONE
;    90          0183  1 !
;    91          0184  1 ! PSECTS:
;    92          0185  1 !
;    93          0186  1 DECLARE_PSECTS (BAS);                       ! Declare psects for BAS$ facility
;    94          0187  1 !
;    95          0188  1 ! OWN STORAGE:
;    96          0189  1 !
;    97          0190  1 !       NONE
;    98          0191  1 !
;    99          0192  1 ! EXTERNAL REFERENCES:
;   100          0193  1 !
;   101          0194  1 EXTERNAL ROUTINE
;   102          0195  1     BAS$$STOP : NOVALUE;                    ! Signal fatal error
;   103          0196  1
;   104          0197  1 EXTERNAL LITERAL
;   105          0198  1     BAS$K_ARGDONMAT : UNSIGNED (8),
;   106          0199  1     BAS$K_NOTIMP : UNSIGNED (8),
;   107          0200  1     BAS$K_SUBOUTRAN : UNSIGNED (8),
;   108          0201  1     BAS$K_TOOFEWARG : UNSIGNED (8),
;   109          0202  1     BAS$K_TOOMANARG : UNSIGNED (8);
;   110          0203  1
;   111          0204  1
```

```
;  113          0205  1 GLOBAL ROUTINE BAS$FETCH_ADDR (                      ! Fetch address of array element
;  114          0206  1         DESCRIP,                                     ! The descriptor
;  115          0207  1         INDEX1                                       ! First index
;  116          0208  1     ) : =
;  117          0209  1
;  118          0210  1 !++
;  119          0211  1 ! FUNCTIONAL DESCRIPTION:
;  120          0212  1 !
;  121          0213  1 !       Given a descriptor for the array and the indices, calculate
;  122          0214  1 !       the address of an element.  Take into account that this may
;  123          0215  1 !       be a FORTRAN array.  This routine does not handle virtual
;  124          0216  1 !       arrays.
;  125          0217  1 !
;  126          0218  1 ! FORMAL PARAMETERS:
;  127          0219  1 !
;  128          0220  1 !       DESCRIP.rx.da   The descriptor of the array
;  129          0221  1 !       INDEX1.rl.v     The first index into the array.  More indicies
;  130          0222  1 !                       may follow this one in the calling sequence.
;  131          0223  1 !
;  132          0224  1 ! IMPLICIT INPUTS:
;  133          0225  1 !
;  134          0226  1 !       NONE
;  135          0227  1 !
;  136          0228  1 ! IMPLICIT OUTPUTS:
;  137          0229  1 !
;  138          0230  1 !       NONE
;  139          0231  1 !
;  140          0232  1 ! ROUTINE VALUE:
;  141          0233  1 !
;  142          0234  1 !       The address of the element is returned
;  143          0235  1 !
;  144          0236  1 ! COMPLETION CODES:
;  145          0237  1 !
;  146          0238  1 !       NONE
;  147          0239  1 !
;  148          0240  1 ! SIDE EFFECTS:
;  149          0241  1 !
;  150          0242  1 !       Signals if an error is encountered.
;  151          0243  1 !
;  152          0244  1 !--
;  153          0245  1
;  154          0246  2     BEGIN
;  155          0247  2
;  156          0248  2     BUILTIN
;  157          0249  2         ACTUALCOUNT,
;  158          0250  2         ACTUALPARAMETER;
;  159          0251  2
;  160          0252  2     LOCAL
;  161          0253  2         INDEX_VALUE,
;  162          0254  2         VALUE_LOCATION,
;  163          0255  2         MULTIPLIERS : REF VECTOR,
;  164          0256  2         BOUNDS : REF VECTOR,
;  165          0257  2         LOW_INDEX,
;  166          0258  2         HIGH_INDEX,
;  167          0259  2         INDEX_INCR,
;  168          0260  2         INDEX_NUMBER,
;  169          0261  2         VALUE_ADDR,
```

```
;  170          0262  2          LENGTH;
;  171          0263
;  172          0264  2      MAP
;  173          0265  2          DESCRIP : REF BLOCK [8, BYTE];
;  174          0266
;  175          0267  2  !+
;  176          0268  2  ! Be sure the number of array subscripts matches the number of
;  177          0269  2  ! indicies given to us.
;  178          0270  2  !-
;  179          0271
;  180          0272  2      IF ((ACTUALCOUNT () - 1) NEQU .DESCRIP [DSC$B_DIMCT])
;  181          0273  2      THEN
;  182          0274  3          BEGIN
;  183          0275
;  184          0276  4          IF ((ACTUALCOUNT () - 1) LSSU .DESCRIP [DSC$B_DIMCT])
;  185          0277  4          THEN
;  186          0278  4              BAS$$STOP (BAS$K_TOOFEWARG)
;  187          0279  4          ELSE
;  188          0280  4              BAS$$STOP (BAS$K_TOOMANARG);
;  189          0281
;  190          0282  3          END;
;  191          0283
;  192          0284  2  !+
;  193          0285  2  ! The coefficients and bounds must be present.
;  194          0286  2  !-
;  195          0287  2
;  196          0288  2      IF ( NOT (.DESCRIP [DSC$V_FL_COEFF] AND .DESCRIP [DSC$V_FL_BOUNDS])) THEN BAS$$STOP (BAS$K_ARGDONMAT);
;  197          0289  2
;  198          0290  2      MULTIPLIERS = DESCRIP [DSC$L_M1];
;  199          0291  2      BOUNDS = DESCRIP [DSC$L_M1] + (%UPVAL*.DESCRIP [DSC$B_DIMCT]);
;  200          0292  2  !+
;  201          0293  2  ! Compute the lower and upper index numbers based on how the array
;  202          0294  2  ! is stored.
;  203          0295  2  !-
;  204          0296  2
;  205          0297  3      IF (.DESCRIP [DSC$V_FL_COLUMN])
;  206          0298  2      THEN
;  207          0299  3          BEGIN
;  208          0300  3          LOW_INDEX = .DESCRIP [DSC$B_DIMCT];
;  209          0301  3          HIGH_INDEX = 1;
;  210          0302  3          INDEX_INCR = -1;
;  211          0303  3          END
;  212          0304  2      ELSE
;  213          0305  3          BEGIN
;  214          0306  3          LOW_INDEX = 1;
;  215          0307  3          HIGH_INDEX = .DESCRIP [DSC$B_DIMCT];
;  216          0308  3          INDEX_INCR = 1;
;  217          0309  2          END;
;  218          0310  2
;  219          0311  2      INDEX_NUMBER = .LOW_INDEX - .INDEX_INCR;
;  220          0312  2  !+
;  221          0313  2  ! Recompute decimal length if necessary.
;  222          0314  2  !-
;  223          0315  2      IF .DESCRIP [DSC$B_DTYPE] EQL DSC$K_DTYPE_P
;  224          0316  2      THEN
;  225          0317  2          LENGTH = .DESCRIP [DSC$W_LENGTH]/2 + 1
;  226          0318  2      ELSE
```

```
;  227          0319  2          LENGTH = .DESCRIP [DSC$W_LENGTH];
;  228          0320  2      !+
;  229          0321  2      ! Compute the linear index from the indices provided.
;  230          0322  2      !-
;  231          0323  2          VALUE_LOCATION = 0;
;  232          0324  2
;  233          0325  2          WHILE ((INDEX_NUMBER = .INDEX_NUMBER + .INDEX_INCR) NEQ (.HIGH_INDEX + .INDEX_INCR)) DO
;  234          0326  2              BEGIN
;  235          0327  2              INDEX_VALUE = ACTUALPARAMETER (.INDEX_NUMBER + 1);
;  236          0328  2
;  237          0329  2              IF ((.INDEX_VALUE LSS .BOUNDS [(.INDEX_NUMBER - 1)*2])  !
;  238          0330  2                  OR (.INDEX_VALUE GTR .BOUNDS [((.INDEX_NUMBER - 1)*2) + 1]))
;  239          0331  2              THEN
;  240          0332  2                  BAS$$STOP (BAS$K_SUBOUTRAN);
;  241          0333  2
;  242          0334  2              VALUE_LOCATION = (.VALUE_LOCATION*.MULTIPLIERS [.INDEX_NUMBER - 1]) + .INDEX_VALUE;
;  243          0335  2              END;
;  244          0336  2
;  245          0337  2          VALUE_LOCATION = (.VALUE_LOCATION*.LENGTH) + .DESCRIP [DSC$A_A0];
;  246          0338  2
;  247          0339  2      !+
;  248          0340  2      ! Check for an array of descriptors.  Fetch the address from the pointer
;  249          0341  2      ! field of the descriptor if necessary.
;  250          0342  2      !-
;  251          0343  2
;  252          0344  2          IF (.DESCRIP [DSC$B_DTYPE] EQLU DSC$K_DTYPE_DSC)
;  253          0345  2          THEN
;  254          0346  2              BEGIN
;  255          0347  2
;  256          0348  2              MAP
;  257          0349  2                  VALUE_LOCATION : REF BLOCK [8, BYTE];
;  258          0350  2
;  259          0351  2              IF .VALUE_LOCATION [DSC$B_DTYPE] NEQ DSC$K_DTYPE_T
;  260          0352  2              THEN
;  261          0353  2                  VALUE_ADDR = .VALUE_LOCATION [DSC$A_POINTER]
;  262          0354  2              ELSE
;  263          0355  2                  VALUE_ADDR = .VALUE_LOCATION;
;  264          0356  2              END
;  265          0357  2          ELSE
;  266          0358  2              BEGIN
;  267          0359  2              VALUE_ADDR = .VALUE_LOCATION;
;  268          0360  2              END;
;  269          0361  2
;  270          0362  2
;  271          0363  2          IF (.DESCRIP [DSC$B_CLASS] NEQU DSC$K_CLASS_A) THEN BAS$$STOP (BAS$K_NOTIMP);
;  272          0364  2
;  273          0365  2          RETURN .VALUE_ADDR;
;  274          0366  2
;  275          0367  1          END;                                    ! end of BAS$FETCH_ADDR

                                                    .TITLE  BAS$FETCH_ADDR
                                                    .IDENT  \1-004\

                                                    .EXTRN  BAS$$STOP, BAS$K_ARGDONMAT
                                                    .EXTRN  BAS$K_NOTIMP, BAS$K_SUBOUTRAN
                                                    .EXTRN  BAS$K_TOOFEWARG
```

```
                                                        .EXTRN  BAS$K_TOOMANARG

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

                                        0FFC 00000      .ENTRY  BAS$FETCH_ADDR, Save R2,R3,R4,R5,R6,R7,R8,-      ; 0205
                                                                R9,R10,R11
                    5B 00000000G    00  9E 00002      MOVAB   BAS$$STOP, R11
                    50              6C  9A 00009      MOVZBL  (AP), R0                                          ; 0272
                                    50  D7 0000C      DECL    R0
                    56        04    AC  D0 0000E      MOVL    DESCRIP, R6
                    52        0B    A6  9A 00012      MOVZBL  11(R6), R2
                    52              50  D1 00016      CMPL    R0, R2
                                    17  13 00019      BEQL    3$
                    50              6C  9A 0001B      MOVZBL  (AP), R0                                          ; 0276
                                    50  D7 0001E      DECL    R0
                    52              50  D1 00020      CMPL    R0, R2
                                    06  1E 00023      BGEQU   1$
                    7E        00G   8F  9A 00025      MOVZBL  #BAS$K_TOOFEWARG, -(SP)                           ; 0278
                                    04  11 00029      BRB     2$
                    7E        00G   8F  9A 0002B 1$:  MOVZBL  #BAS$K_TOOMANARG, -(SP)                           ; 0280
                    6B              01  FB 0002F 2$:  CALLS   #1, BAS$$STOP
          05        0A    A6          06  E1 00032 3$:  BBC     #6, 10(R6), 4$                                  ; 0288
                          0A    A6    95  00037       TSTB    10(R6)
                                07  19  0003A       BLSS    5$
                    7E        00G   8F  9A 0003C 4$:  MOVZBL  #BAS$K_ARGDONMAT, -(SP)
                    6B              01  FB 00040       CALLS   #1, BAS$$STOP
                    55        14    A6  9E 00043 5$:  MOVAB   20(R6), MULTIPLIERS                             ; 0290
                    57        14 A642   DE 00047       MOVAL   20(R6)[R2], BOUNDS                              ; 0291
          0B        0A    A6          05  E1 0004C       BBC     #5, 10(R6), 6$                                  ; 0297
                    51              52  D0 00051       MOVL    R2, LOW_INDEX                                   ; 0300
                    50              01  D0 00054       MOVL    #1, HIGH_INDEX                                  ; 0301
                    58              01  CE 00057       MNEGL   #1, INDEX_INCR                                  ; 0302
                                    09  11 0005A       BRB     7$                                              ; 0297
                    51              01  D0 0005C 6$:  MOVL    #1, LOW_INDEX                                   ; 0306
                    50              52  D0 0005F       MOVL    R2, HIGH_INDEX                                  ; 0307
                    58              01  D0 00062       MOVL    #1, INDEX_INCR                                  ; 0308
          52        51              58  C3 00065 7$:  SUBL3   INDEX_INCR, LOW_INDEX, INDEX_NUMBER             ; 0311
                    15        02    A6  91 00069       CMPB    2(R6), #21                                      ; 0315
                                    0A  12 0006D       BNEQ    8$
                    54              66  3C 0006F       MOVZWL  (R6), R4                                        ; 0317
                    54              02  C6 00072       DIVL2   #2, R4
                                    54  D6 00075       INCL    LENGTH
                                    03  11 00077       BRB     9$
                    54              66  3C 00079 8$:  MOVZWL  (R6), LENGTH                                    ; 0319
                                    53  D4 0007C 9$:  CLRL    VALUE_LOCATION                                  ; 0323
          5A        50              58  C1 0007E       ADDL3   INDEX_INCR, HIGH_INDEX, R10                     ; 0325
                    52              58  C0 00082 10$:  ADDL2   INDEX_INCR, INDEX_NUMBER
                    5A              52  D1 00085       CMPL    INDEX_NUMBER, R10
                                    2A  13 00088       BEQL    13$
                    59        04 AC42   D0 0008A       MOVL    4(AP)[INDEX_NUMBER], INDEX_VALUE                ; 0327
          50        52              01  78 0008F       ASHL    #1, INDEX_NUMBER, R0                            ; 0329
                 F8 A740              59  D1 00093       CMPL    INDEX_VALUE, -8(BOUNDS)[R0]
                                    07  19 00098       BLSS    11$
                 FC A740              59  D1 0009A       CMPL    INDEX_VALUE, -4(BOUNDS)[R0]                     ; 0330
                                    07  15 0009F       BLEQ    12$
                    7E        00G   8F  9A 000A1 11$:  MOVZBL  #BAS$K_SUBOUTRAN, -(SP)                        ; 0332
                    6B              01  FB 000A5       CALLS   #1, BAS$$STOP
```

```
                50              53      FC A542  C5 000A8 12$:    MULL3   -4(MULTIPLIERS)[INDEX_NUMBER], -        ; 0334
                                                                         VALUE_LOCATION, R0
                53              50          59   C1 000AE         ADDL3   INDEX_VALUE, R0, VALUE_LOCATION
                                            CE   11 000B2         BRB     10$                                     ; 0325
                50              53          54   C5 000B4 13$:    MULL3   LENGTH, VALUE_LOCATION, R0              ; 0337
                53              50      10  A6   C1 000B8         ADDL3   16(R6), R0, VALUE_LOCATION
                                18      02  A6   91 000BD         CMPB    2(R6), #24                              ; 0344
                                            0C   12 000C1         BNEQ    14$
                                0E      02  A3   91 000C3         CMPB    2(VALUE_LOCATION), #14                  ; 0351
                                            06   13 000C7         BEQL    14$
                                52      04  A3   D0 000C9         MOVL    4(VALUE_LOCATION), VALUE_ADDR           ; 0353
                                            03   11 000CD         BRB     15$
                                52          53   D0 000CF 14$:    MOVL    VALUE_LOCATION, VALUE_ADDR              ; 0359
                                04      03  A6   91 000D2 15$:    CMPB    3(R6), #4                               ; 0363
                                            07   13 000D6         BEQL    16$
                                7E      00G 8F   9A 000D8         MOVZBL  #BAS$K_NOTIMP, -(SP)
                                6B          01   FB 000DC         CALLS   #1, BAS$$STOP
                                50          52   D0 000DF 16$:    MOVL    VALUE_ADDR, R0                          ; 0365
                                                 04 000E2         RET                                             ; 0367

; Routine Size:  227 bytes,    Routine Base:  _BAS$CODE + 0000

;  276          0368  1
;  277          0369  1 END                                     ! end of module BAS$FETCH_ADDR
;  278          0370  1
;  279          0371  0 ELUDOM
```

```
;                               PSECT SUMMARY

;     Name                      Bytes                         Attributes

;  _BAS$CODE                     227  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        14         0         581         00:01.1


;                               COMMAND QUALIFIERS
```

```
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BASFETCHA/OBJ=OBJ$:BASFETCHA MSRC$:BASFETCHA/UPDATE=(ENH$:BASFETCHA
;       )

; Size:         227 code + 0 data bytes
; Run Time:        00:07.2
; Elapsed Time:    00:16.2
; Lines/CPU Min:    3078
; Lexemes/CPU-Min: 15585
; Memory Used:  103 pages
; Compilation Complete
```

BASFREE
LIS

BASEXITHA
LIS

BASFETCHD
LIS

BASFORINI
LIS

BASGETRFA
LIS

BASFETCHA
LIS

BASGET
LIS

BASFSP
LIS

BASFIND
LIS

BASFORMAT
LIS

BASHANDLE
LIS