```
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  RRRRRRRR    TTTTTTTTTT  DDDDDDDD      IIIIII    MM      MM
BBBBBBBB      AAAAAA      SSSSSSSS  RRRRRRRR    TTTTTTTTTT  DDDDDDDD      IIIIII    MM      MM
BB      BB   AA      AA  SS        RR      RR      TT      DD      DD      II      MMMM  MMMM
BB      BB   AA      AA  SS        RR      RR      TT      DD      DD      II      MMMM  MMMM
BB      BB   AA      AA  SS        RR      RR      TT      DD      DD      II      MM  MM  MM
BB      BB   AA      AA  SS        RR      RR      TT      DD      DD      II      MM  MM  MM
BBBBBBBB     AA      AA    SSSSSS  RRRRRRRR        TT      DD      DD      II      MM      MM
BBBBBBBB     AA      AA    SSSSSS  RRRRRRRR        TT      DD      DD      II      MM      MM
BB      BB   AAAAAAAAAA          SS  RR  RR        TT      DD      DD      II      MM      MM
BB      BB   AAAAAAAAAA          SS  RR  RR        TT      DD      DD      II      MM      MM
BB      BB   AA      AA          SS  RR    RR      TT      DD      DD      II      MM      MM      ....
BB      BB   AA      AA          SS  RR    RR      TT      DD      DD      II      MM      MM      ....
BBBBBBBB     AA      AA  SSSSSSSS    RR      RR      TT      DDDDDDDD      IIIIII    MM      MM      ....
BBBBBBBB     AA      AA  SSSSSSSS    RR      RR      TT      DDDDDDDD      IIIIII    MM      MM      ....

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$RT_DIM (                                  ! Run time dimension
;     2          0002  0                 IDENT = '1-011'                     ! File: BASRTDIM.B32 Edit: MDL1011
;     3          0003  0                 ) =
;     4          0004  0 !
;     5          0005  0 !*****************************************************************************
;     6          0006  0 !*                                                                           *
;     7          0007  0 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;     8          0008  0 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;     9          0009  0 !*  ALL RIGHTS RESERVED.                                                     *
;    10          0010  0 !*                                                                           *
;    11          0011  0 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    12          0012  0 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    13          0013  0 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    14          0014  0 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    15          0015  0 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    16          0016  0 !*  TRANSFERRED.                                                             *
;    17          0017  0 !*                                                                           *
;    18          0018  0 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    19          0019  0 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    20          0020  0 !*  CORPORATION.                                                             *
;    21          0021  0 !*                                                                           *
;    22          0022  0 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    23          0023  0 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    24          0024  0 !*                                                                           *
;    25          0025  0 !*                                                                           *
;    26          0026  0 !*****************************************************************************
;    27          0027  0 !
;    28          0028  1 BEGIN
;    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 routine is called by the generated code to dimension an
;    36          0036  1 !       array at run-time.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  Runs at any access mode - AST reentrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: Pamela L. Levesque, CREATION DATE: 14-May-1982
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 Original.  PLL 14-May-1982
;    45          0045  1 ! 1-002 Allow an upper bound of 0.  PLL 20-May-1982
;    46          0046  1 ! 1-003 Make routine global.  PLL 3-Jun-1982
;    47          0047  1 ! 1-004 Check return status from LIB$FREE_VM and LIB$GET_VM.  PLL 7-Jun-1982
;    48          0048  1 ! 1-005 Row 0 should be included in ARSIZE.  Also, the counter in the INCR
;    49          0049  1 !       loop for setting the bounds should start at 0, not 1.  PLL 30-Jun-1982
;    50          0050  1 ! 1-006 Initialize BNDSARG outside of the loop which sets the lower and
;    51          0051  1 !       upper bounds.  PLL 27-Sep-1982
;    52          0052  1 ! 1-007 Fix initialization of array elements to not write beyond the space
;    53          0053  1 !       allocated by LIB$GET_VM.  (End_addr was incorrectly computed.)
;    54          0054  1 !       PLL 1-Oct-1982
;    55          0055  1 ! 1-008 - Add RECORD datatype.  Since there is no datatype by this name on the
;    56          0056  1 !         VAX, Basic uses DSC$K_DTYPE_Z (none).  MDL 26-May-1983
;    57          0057  1 ! 1-009 - correct array initialization.  use a loop of CH$FILL's; the MOVC5
```

```
;    58          0058  1 !           generated by one CH$x comes nowhere near writing all of a large array,
;    59          0059  1 !           as the length operands for the instruction are words.  MDL 20-Dec-1983
;    60          0060  1 ! 1-010 -   the size to return to FREE_VM should be calculated from the array size
;    61          0061  1 !           field in the array descriptor, in the same fashion as for the call to
;    62          0062  1 !           GET_VM when the array is first allocated.  MDL 9-Feb-1984
;    63          0063  1 ! 1-011 -   check to see if there's enough virtual address space available before
;    64          0064  1 !           calling GET_VM.  also, free dynamic strings if returning an array of
;    65          0065  1 !           them.  MDL 22-Feb-1984
;    66          0066  1 !--
;    67          0067  1
```

```
;    69          0068  1 %SBTTL 'Declarations'
;    70          0069  1 !
;    71          0070  1 ! SWITCHES:
;    72          0071  1 !
;    73          0072  1
;    74          0073  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    75          0074  1
;    76          0075  1 !
;    77          0076  1 ! LINKAGES:
;    78          0077  1 !
;    79          0078  1 !       NONE
;    80          0079  1 !
;    81          0080  1 ! TABLE OF CONTENTS:
;    82          0081  1 !
;    83          0082  1
;    84          0083  1 FORWARD ROUTINE
;    85          0084  1     BAS$RT_DIM : NOVALUE;                       ! run time dimension
;    86          0085  1
;    87          0086  1 !
;    88          0087  1 ! INCLUDE FILES:
;    89          0088  1 !
;    90          0089  1
;    91          0090  1 LIBRARY 'RTLSTARLE';                            ! System symbols, typically from SYS$LIBRARY:STARLET.L32
;    92          0091  1
;    93          0092  1 REQUIRE 'RTLIN:RTLPSECT';                       ! Define PSECT declarations macros
;    94          0187  1
;    95          0188  1 !
;    96          0189  1 ! MACROS:
;    97          0190  1 !
;    98          0191  1 !       NONE
;    99          0192  1 !
;   100          0193  1 ! EQUATED SYMBOLS:
;   101          0194  1 !
;   102          0195  1 !       NONE
;   103          0196  1 !
;   104          0197  1 ! FIELDS:
;   105          0198  1 !
;   106          0199  1 !       NONE
;   107          0200  1 !
;   108          0201  1 ! PSECTS:
;   109          0202  1 !
;   110          0203  1 DECLARE_PSECTS (BAS);                           ! Declare PSECTs for BAS$ facility
;   111          0204  1
;   112          0205  1 ! OWN STORAGE:
;   113          0206  1 !
;   114          0207  1 !       NONE
;   115          0208  1 !
;   116          0209  1 ! EXTERNAL REFERENCES:
;   117          0210  1 !
;   118          0211  1
;   119          0212  1 EXTERNAL ROUTINE
;   120          0213  1     BAS$$STOP : NOVALUE,                        ! Signal a fatal error
;   121          0214  1     LIB$GET_EF,                                 ! get an event flag
;   122          0215  1     LIB$FREE_EF,                                ! free an event flag
;   123          0216  1     LIB$STOP : NOVALUE,                         ! signal if error from GET_EF or $GETJPI
;   124          0217  1     LIB$FREE_VM,                                ! Deallocate virtual space
;   125          0218  1     LIB$GET_VM,                                 ! Allocate virtual space
```

```
;  126          0219  1     STR$FREE1_DX;                          ! Deallocate string space
;  127          0220  1
;  128          0221  1 EXTERNAL LITERAL
;  129          0222  1     BAS$K_ARGDONMAT : UNSIGNED (8),        ! Condition value symbols
;  130          0223  1     BAS$K_NOTIMP : UNSIGNED (8),           ! Arguments don't match
;  131          0224  1     BAS$K_SUBOUTRAN : UNSIGNED (8),        ! Not implemented
;  132          0225  1     BAS$K_TOOFEWARG : UNSIGNED (8),        ! Subscript out of range
;  133          0226  1     BAS$K_TOOMANARG : UNSIGNED (8),        ! Too few arguments
;  134          0227  1     BAS$K_MAXMEMEXC : UNSIGNED (8),        ! Too many arguments
;  135          0228  1     BAS$K_PROLOSSOR : UNSIGNED (8);        ! Maximum memory exceeded
;  136          0229  1                                            ! Program lost sorry
```

```
;  138     0230  1 %SBTTL 'BAS$RT_DIM - run time dimension'
;  139     0231  1 GLOBAL ROUTINE BAS$RT_DIM (                      ! run time dimension
;  140     0232  1         ARRAY,                                   ! array to re-dim
;  141     0233  1         BOUND1                                   ! 1st upper bound
;  142     0234  1     ) : NOVALUE =
;  143     0235  1
;  144     0236  1 !++
;  145     0237  1 ! FUNCTIONAL DESCRIPTION:
;  146     0238  1 !
;  147     0239  1 !       This routine changes the size of an array.  It does this by
;  148     0240  1 !       deleting any virtual memory previously allocated for the array
;  149     0241  1 !       elements, updating the upper bounds with the parameters passed,
;  150     0242  1 !       and then allocating enough virtual memory for the new size.
;  151     0243  1 !
;  152     0244  1 !       Numeric array elements are initialized to zero, and string array
;  153     0245  1 !       elements (descriptors) have their length and pointer set to zero.
;  154     0246  1 !
;  155     0247  1 ! CALLING SEQUENCE:
;  156     0248  1 !
;  157     0249  1 !       BAS$RT_DIM (ARRAY.mx.da, BOUND1.rl.v [,BOUND2.rl.v, ... BOUND32.rl.v])
;  158     0250  1 !
;  159     0251  1 ! FORMAL PARAMETERS:
;  160     0252  1 !
;  161     0253  1 !       ARRAY           desc. of array to re-dimension
;  162     0254  1 !       BOUND1          new value of 1st upper bound
;  163     0255  1 !       [BOUND2,...     optional parameters - the number of bounds
;  164     0256  1 !         BOUND32]      passed must agree with the number of dimensions
;  165     0257  1 !                       up to 32 dimensions are possible
;  166     0258  1 !
;  167     0259  1 ! IMPLICIT INPUTS:
;  168     0260  1 !
;  169     0261  1 !       NONE
;  170     0262  1 !
;  171     0263  1 ! IMPLICIT OUTPUTS:
;  172     0264  1 !
;  173     0265  1 !       The following fields in the array descriptor are updated:
;  174     0266  1 !               DSC$A_POINTER
;  175     0267  1 !               DSC$A_A0
;  176     0268  1 !               DSC$L_ARSIZE
;  177     0269  1 !               bounds and multipliers
;  178     0270  1 !
;  179     0271  1 ! COMPLETION STATUS:
;  180     0272  1 !
;  181     0273  1 !       Signals if any error is encountered
;  182     0274  1 !
;  183     0275  1 ! SIDE EFFECTS:
;  184     0276  1 !
;  185     0277  1 !       NONE
;  186     0278  1 !
;  187     0279  1 !--
;  188     0280  1
;  189     0281  2     BEGIN
;  190     0282  2
;  191     0283  2     BUILTIN
;  192     0284  2         ACTUALCOUNT,
;  193     0285  2         ACTUALPARAMETER,
;  194     0286  2         ASHP;
```

```
;  195     0287  2
;  196     0288  2      MAP
;  197     0289  2          ARRAY : REF BLOCK [,BYTE];
;  198     0290  2
;  199     0291  2      LOCAL
;  200     0292  2          AR_SIZE : INITIAL (1),                  ! temp used to compute array size
;  201     0293  2          BNDSARG : INITIAL (2),                  ! 1st bounds is 2nd arg
;  202     0294  2          BOUNDS,                                 ! upper or lower bounds of array
;  203     0295  2          MULTIPLIERS,                            ! mult in array desc
;  204     0296  2          LENGTH,                                 ! temp used to compute element size
;  205     0297  2          STATUS,                                 ! return status from calls
;  206     0298  2          EVENT_FLAG,                             ! for use with $GETJPI
;  207     0299  2          NUM_FREE_PAGES: INITIAL(0),             ! written by $GETJPI
;  208     0300  2          JPI_RETURN_LENGTH: INITIAL(0),          ! # bytes returned from $GETJPI
;  209     0301  2          JPI_ITEMS: VECTOR [4, LONG] INITIAL (   ((JPI$_FREPTECNT^16) OR 4),
;  210     0302  2                                                  NUM_FREE_PAGES,
;  211     0303  2                                                  JPI_RETURN_LENGTH,
;  212     0304  2                                                  0 );
;  213     0305  2
;  214     0306  2  !+
;  215     0307  2  ! First perform some checks to make sure the proper number of bounds have
;  216     0308  2  ! been passed, etc.
;  217     0309  2  !-
;  218     0310  2
;  219     0311  2      IF (ACTUALCOUNT () - 1) NEQ .ARRAY [DSC$B_DIMCT]
;  220     0312  2      THEN
;  221     0313  3          BEGIN
;  222     0314  3          IF (ACTUALCOUNT () - 1) LSS .ARRAY [DSC$B_DIMCT]
;  223     0315  3          THEN
;  224     0316  3              BAS$$STOP (BAS$K_TOOFEWARG)
;  225     0317  3          ELSE
;  226     0318  3              BAS$$STOP (BAS$K_TOOMANARG);
;  227     0319  2          END;
;  228     0320  2
;  229     0321  3      IF (NOT (.ARRAY [DSC$V_FL_COEFF] AND .ARRAY [DSC$V_FL_BOUNDS]))
;  230     0322  2      THEN
;  231     0323  2          BAS$$STOP (BAS$K_ARGDONMAT);            ! bounds and mult must be present
;  232     0324  2
;  233     0325  2  !+
;  234     0326  2  ! Free previously used space for elements, if any.
;  235     0327  2  !-
;  236     0328  2      IF.ARRAY [DSC$A_POINTER] NEQ 0
;  237     0329  2      THEN
;  238     0330  3          BEGIN
;  239     0331  3          !+
;  240     0332  3          ! if this was an array of descriptors, there is some cleanup necessary
;  241     0333  3          ! prior to freeing the space for the array itself.
;  242     0334  3          !
;  243     0335  3          ! for dynamic string descriptors, each descriptor with space allocated
;  244     0336  3          ! must be freed.
;  245     0337  3          ! a static string descriptor implies that the element is FIELDed,
;  246     0338  3          ! and the FIELD variable must be marked invalid.
;  247     0339  3          !-
;  248     0340  3          IF .ARRAY [DSC$B_DTYPE] EQL DSC$K_DTYPE_DSC
;  249     0341  3          THEN
;  250     0342  4              BEGIN
;  251     0343  4              LOCAL
```

```
;  252     0344  4                END_ADDR;
;  253     0345  4            END_ADDR = .ARRAY [DSC$A_POINTER] + .ARRAY [DSC$L_ARSIZE] -
;  254     0346  4                .ARRAY [DSC$W_LENGTH];
;  255     0347  4            !+
;  256     0348  4            ! walk through the descriptors one at a time.
;  257     0349  4            !-
;  258     0350  4            INCR DSC_ADDR FROM .ARRAY [DSC$A_POINTER] TO .END_ADDR BY .ARRAY [DSC$W_LENGTH] DO
;  259     0351  5                BEGIN
;  260     0352  5                MAP
;  261     0353  5                   DSC_ADDR : REF BLOCK [8,BYTE];
;  262     0354  5
;  263     0355  5                !+
;  264     0356  5                ! dynamic? call STR$FREE1_DX.
;  265     0357  5                !-
;  266     0358  5                IF .DSC_ADDR [DSC$B_CLASS] EQL DSC$K_CLASS_D
;  267     0359  5                THEN
;  268     0360  6                    BEGIN
;  269     0361  6                    STATUS = STR$FREE1_DX (.DSC_ADDR);
;  270     0362  5                    END;
;  271     0363  5
;  272     0364  5                !+
;  273     0365  5                ! static?  mark length as 0 to prevent illicit usage as
;  274     0366  5                ! a FIELDed variable.
;  275     0367  5                !-
;  276     0368  5                IF .DSC_ADDR [DSC$B_CLASS] EQL DSC$K_CLASS_S
;  277     0369  5                THEN
;  278     0370  6                    BEGIN
;  279     0371  6                    DSC_ADDR [DSC$W_LENGTH] = 0;
;  280     0372  5                    END;
;  281     0373  5
;  282     0374  4                END;    ! Incr dsc_addr
;  283     0375  4
;  284     0376  3            END;    ! If .array [dsc$b_dtype] eql dsc$k_dtype_dsc
;  285     0377  3
;  286     0378  3            !+
;  287     0379  3            ! finally, free the old array itself up.
;  288     0380  3            !-
;  289     0381  4            LENGTH = (IF .ARRAY [DSC$B_DTYPE] EQL DSC$K_DTYPE_P
;  290     0382  4                  THEN
;  291     0383  4                    (.ARRAY [DSC$L_ARSIZE]/ .ARRAY [DSC$W_LENGTH]) *
;  292     0384  5                    (.ARRAY [DSC$W_LENGTH]/ 2 + 1)  ! compute length in bytes
;  293     0385  4                  ELSE
;  294     0386  3                    .ARRAY [DSC$L_ARSIZE]);
;  295     0387  3            STATUS = LIB$FREE_VM ( LENGTH, ARRAY [DSC$A_POINTER]);
;  296     0388  4            IF NOT (.STATUS)
;  297     0389  3            THEN
;  298     0390  3                BAS$$STOP (BAS$K_PROLOSSOR);
;  299     0391  3
;  300     0392  2            END;    ! If .array [dsc$a_pointer] neq 0
;  301     0393  2
;  302     0394  2 !+
;  303     0395  2 ! Calculate the amount of space needed for the new size.
;  304     0396  2 !-
;  305     0397  2
;  306     0398  2        INCR COUNTER FROM 1 TO .ARRAY [DSC$B_DIMCT] DO
;  307     0399  3            BEGIN
;  308     0400  3            IF ACTUALPARAMETER (.COUNTER + 1) LSS 0
```

```
;  309          0401  3          THEN
;  310          0402  3              BAS$$STOP (BAS$K_SUBOUTRAN);          ! don't allow bounds < 0
;  311          0403  3          AR_SIZE = .AR_SIZE * (ACTUALPARAMETER (.COUNTER + 1) + 1);
;  312          0404  2          END;
;  313          0405  2
;  314          0406  2      AR_SIZE = .AR_SIZE * .ARRAY [DSC$W_LENGTH];
;  315          0407  2      ARRAY [DSC$L_ARSIZE] = .AR_SIZE;
;  316          0408  2
;  317          0409  2  !+
;  318          0410  2  ! Update the bounds and multipliers in the array descriptor.
;  319          0411  2  !-
;  320          0412  2
;  321          0413  2      BOUNDS = ARRAY [DSC$L_M1] + (%UPVAL * .ARRAY [DSC$B_DIMCT]);
;  322          0414  2      MULTIPLIERS = ARRAY [DSC$L_M1];
;  323          0415  2
;  324          0416  2      INCR COUNTER FROM 0 TO ((.ARRAY [DSC$B_DIMCT] * 2)  - 1) BY 2 DO
;  325          0417  3          BEGIN
;  326          0418  3          MAP
;  327          0419  3              BOUNDS : REF VECTOR;
;  328          0420  3
;  329          0421  3          BOUNDS [.COUNTER] = 0;                    ! set lower bounds to 0
;  330          0422  3          BOUNDS [.COUNTER + 1] = ACTUALPARAMETER (.BNDSARG);
;  331          0423  3                                                    ! take upper bound that was passed
;  332          0424  3          BNDSARG = .BNDSARG + 1;                   ! point to next bounds arg, if any
;  333          0425  3
;  334          0426  2          END;
;  335          0427  2
;  336          0428  2      INCR COUNTER FROM 0 TO (.ARRAY [DSC$B_DIMCT] - 1) DO
;  337          0429  3          BEGIN
;  338          0430  3          MAP
;  339          0431  3              MULTIPLIERS : REF VECTOR;
;  340          0432  3          MULTIPLIERS [.COUNTER] = ACTUALPARAMETER (.COUNTER + 2) + 1;
;  341          0433  2          END;
;  342          0434  2
;  343          0435  2  !+
;  344          0436  2  ! compute the total amount of space needed for the array, in bytes.
;  345          0437  2  !-
;  346          0438  3      LENGTH = (IF .ARRAY [DSC$B_DTYPE] EQL DSC$K_DTYPE_P
;  347          0439  3                THEN
;  348          0440  3                  (.ARRAY [DSC$L_ARSIZE]/ .ARRAY [DSC$W_LENGTH]) *
;  349          0441  4                  (.ARRAY [DSC$W_LENGTH]/ 2 + 1)  ! compute length in bytes
;  350          0442  3                ELSE
;  351          0443  2                  .ARRAY [DSC$L_ARSIZE]);
;  352          0444  2
;  353          0445  2  !+
;  354          0446  2  ! see if there's enough room to expand this process' virtual address space
;  355          0447  2  ! that much.  get the number of free pages available, multiply it by 512
;  356          0448  2  ! to get number of free bytes available, and see if this is more than the
;  357          0449  2  ! number of bytes needed (computed above).
;  358          0450  2  !-
;  359          0451  2      STATUS = LIB$GET_EF (EVENT_FLAG);
;  360          0452  2      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  361          0453  2
;  362          0454  2      STATUS = $GETJPIW ( EFN = .EVENT_FLAG, ITMLST = JPI_ITEMS );
;  363          0455  2      IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  364          0456  2
;  365          0457  2      STATUS = LIB$FREE_EF (EVENT_FLAG);
```

```
;  366     0458  2     IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  367     0459  2
;  368     0460  2     IF (.NUM_FREE_PAGES * 512) LSSU .LENGTH
;  369     0461  2     THEN
;  370     0462  2         BAS$$STOP (BAS$K_MAXMEMEXC);
;  371     0463  2
;  372     0464  2 !+
;  373     0465  2 ! Allocate space for the new size and store it's address in the pointer field
;  374     0466  2 ! of the array descriptor.
;  375     0467  2 !-
;  376     0468  2     STATUS = LIB$GET_VM (LENGTH, ARRAY [DSC$A_POINTER]);
;  377     0469  3     IF NOT (.STATUS)
;  378     0470  2     THEN
;  379     0471  2         BAS$$STOP (BAS$K_MAXMEMEXC);
;  380     0472  2     ARRAY [DSC$A_A0] = .ARRAY [DSC$A_POINTER];
;  381     0473  2
;  382     0474  2 !+
;  383     0475  2 ! Initialize the space according to the data type of the array elements.
;  384     0476  2 !-
;  385     0477  2
;  386     0478  2     SELECTONE .ARRAY [DSC$B_DTYPE] OF
;  387     0479  2         SET
;  388     0480  2
;  389     0481  2         [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F,
;  390     0482  2          DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H, DSC$K_DTYPE_Z] :
;  391     0483  3             BEGIN
;  392     0484  3             LOCAL
;  393     0485  3                 CURR_SIZE:      INITIAL(.ARRAY [DSC$L_ARSIZE]),
;  394     0486  3                 CURR_ADDR:      INITIAL(.ARRAY [DSC$A_POINTER]);
;  395     0487  3
;  396     0488  3             !+
;  397     0489  3             ! initialize the array in chunks of %X'ffff' bytes at a time.  We do this
;  398     0490  3             ! because CH$FILL generates a MOVC5, and the length operand for MOVC5
;  399     0491  3             ! is a WORD.  The total length of the array can be much longer than can
;  400     0492  3             ! be addressed by a WORD.
;  401     0493  3             !-
;  402     0494  3             WHILE .CURR_ADDR LSS (.ARRAY [DSC$A_POINTER] + .ARRAY [DSC$L_ARSIZE]) DO
;  403     0495  4                 BEGIN
;  404     0496  4                 CH$FILL (%C'', MIN ( .CURR_SIZE, %X'0000FFFF'), .CURR_ADDR);
;  405     0497  4                 CURR_SIZE = .CURR_SIZE - %X'0000FFFF';
;  406     0498  4                 CURR_ADDR = .CURR_ADDR + %X'0000FFFF';
;  407     0499  3                 END;
;  408     0500  2             END;
;  409     0501  2
;  410     0502  2         [DSC$K_DTYPE_P] :
;  411     0503  3             BEGIN
;  412     0504  3             LOCAL
;  413     0505  3                 COUNT,
;  414     0506  3                 END_ADDR;
;  415     0507  3             BIND
;  416     0508  3                 PACKED_ZERO = UPLIT BYTE (REP 15 OF (%X'00'), %X'0C');
;  417     0509  3             COUNT = - .ARRAY [DSC$B_SCALE];
;  418     0510  3             END_ADDR = .ARRAY [DSC$A_POINTER] + .LENGTH -
;  419     0511  3                 (.ARRAY [DSC$W_LENGTH]/ 2 + 1);
;  420     0512  3             INCR ELE_PTR FROM .ARRAY [DSC$A_POINTER] TO .END_ADDR
;  421     0513  3                 BY (.ARRAY [DSC$W_LENGTH]/ 2 + 1) DO
;  422     0514  4                 BEGIN
```

```
;  423          0515  4                 ASHP (COUNT, %REF(15), PACKED_ZERO, %REF(0),
;  424          0516  4                      ARRAY [DSC$W_LENGTH], .ELE_PTR);
;  425          0517  3                 END;                                    ! ele by ele copy to
;  426          0518  2             END;                                        ! accomodate scaling
;  427          0519  2
;  428          0520  2             [DSC$K_DTYPE_DSC] :
;  429          0521  3                 BEGIN
;  430          0522  3                 LOCAL
;  431          0523  3                     END_ADDR;
;  432          0524  3                 END_ADDR = .ARRAY [DSC$A_POINTER] + .ARRAY [DSC$L_ARSIZE] -
;  433          0525  3                     .ARRAY [DSC$W_LENGTH];
;  434          0526  3                 INCR DSC_ADDR FROM .ARRAY [DSC$A_POINTER] TO .END_ADDR BY .ARRAY [DSC$W_LENGTH] DO
;  435          0527  4                     BEGIN
;  436          0528  4                     MAP
;  437          0529  4                         DSC_ADDR : REF BLOCK [8,BYTE];
;  438          0530  4                     DSC_ADDR [DSC$B_CLASS] = DSC$K_CLASS_D;
;  439          0531  4                     DSC_ADDR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  440          0532  4                     DSC_ADDR [DSC$W_LENGTH] = 0;
;  441          0533  4                     DSC_ADDR [DSC$A_POINTER] = 0;
;  442          0534  3                     END;
;  443          0535  2             END;
;  444          0536  2
;  445          0537  2             [OTHERWISE] :
;  446          0538  2                 BAS$$STOP (BAS$K_NOTIMP);
;  447          0539  2
;  448          0540  2             TES;
;  449          0541  2
;  450          0542  2
;  451          0543  1     END;                                                ! End of routine BAS$RT_DIM


                                        .TITLE  BAS$RT_DIM
                                        .IDENT  \1-011\

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

                 04150004  00000 P.AAA: .LONG   68485124                                                    ;
                 00000000  00004        .LONG   0                                                           ;
00000000         00000000  00008        .LONG   0, 0                                                        ;
                       00# 00010 P.AAB: .BYTE   0[15]                                                       ;
                       0C  0001F        .BYTE   12                                                          ;

                                PACKED_ZERO=        P.AAB
                                        .EXTRN  BAS$$STOP, LIB$GET_EF
                                        .EXTRN  LIB$FREE_EF, LIB$STOP
                                        .EXTRN  LIB$FREE_VM, LIB$GET_VM
                                        .EXTRN  STR$FREE1_DX, BAS$K_ARGDONMAT
                                        .EXTRN  BAS$K_NOTIMP, BAS$K_SUBOUTRAN
                                        .EXTRN  BAS$K_TOOFEWARG
                                        .EXTRN  BAS$K_TOOMANARG
                                        .EXTRN  BAS$K_MAXMEMEXC
                                        .EXTRN  BAS$K_PROLOSSOR
                                        .EXTRN  SYS$GETJPIW

                       0FFC 00000       .ENTRY  BAS$RT_DIM, Save R2,R3,R4,R5,R6,R7,R8,R9,-                  ; 0231
                                                R10,R11                                                     ;
5B 00000000G     00  9E 00002         MOVAB   BAS$$STOP, R11                                              ;
```

```
                                5E                  1C  C2 00009          SUBL2   #28, SP
                                59                  01  D0 0000C          MOVL    #1, AR_SIZE                                 0281
                                56                  02  D0 0000F          MOVL    #2, BNDSARG
                                                    7E  D4 00012          CLRL    NUM_FREE_PAGES
                                            04      AE  D4 00014          CLRL    JPI_RETURN_LENGTH
                        10  AE      C5      AF      10  28 00017          MOVC3   #16, P.AAA, JPI_ITEMS                       0304
                                        14  AE      6E  9E 0001D          MOVAB   NUM_FREE_PAGES, JPI_ITEMS+4                 0281
                                        18  AE  04  AE  9E 00021          MOVAB   JPI_RETURN_LENGTH, JPI_ITEMS+8
                                50                  6C  9A 00026          MOVZBL  (AP), R0                                    0311
                                                    50  D7 00029          DECL    R0
                                57              04  AC  D0 0002B          MOVL    ARRAY, R7
                50      0B  A7                  08  00  ED 0002F          CMPZV   #0, #8, 11(R7), R0
                                                    1A  13 00035          BEQL    3$
                                50                  6C  9A 00037          MOVZBL  (AP), R0                                    0314
                                                    50  D7 0003A          DECL    R0
                50      0B  A7                  08  00  ED 0003C          CMPZV   #0, #8, 11(R7), R0
                                                    06  15 00042          BLEQ    1$
                                7E          00G 8F  9A 00044          MOVZBL  #BAS$K_TOOFEWARG, -(SP)                     0316
                                                    04  11 00048          BRB     2$
                                7E          00G 8F  9A 0004A 1$:      MOVZBL  #BAS$K_TOOMANARG, -(SP)                     0318
                                6B                  01  FB 0004E 2$:      CALLS   #1, BAS$$STOP
                        05      0A      A7          06  E1 00051 3$:      BBC     #6, 10(R7), 4$                              0321
                                            0A      A7  95 00056          TSTB    10(R7)
                                                    07  19 00059          BLSS    5$
                                7E          00G 8F  9A 0005B 4$:      MOVZBL  #BAS$K_ARGDONMAT, -(SP)                     0323
                                6B                  01  FB 0005F          CALLS   #1, BAS$$STOP
                                58              04  A7  9E 00062 5$:      MOVAB   4(R7), R8                                   0328
                                                    68  D5 00066          TSTL    (R8)
                                                    75  13 00068          BEQL    13$
                                18              02  A7  91 0006A          CMPB    2(R7), #24                                  0340
                                                    33  12 0006E          BNEQ    10$
                54              68              0C  A7  C1 00070          ADDL3   12(R7), (R8), R4                            0345
                                53                  67  3C 00075          MOVZWL  (R7), R3                                    0346
                55              54                  53  C3 00078          SUBL3   R3, R4, END_ADDR
                                52                  68  D0 0007C          MOVL    (R8), DSC_ADDR                              0350
                                                    1D  11 0007F          BRB     9$
                                02              03  A2  91 00081 6$:      CMPB    3(DSC_ADDR), #2                             0358
                                                    0C  12 00085          BNEQ    7$
                                52                  DD  00087          PUSHL   DSC_ADDR                                    0361
                    00000000G   00                  01  FB 00089          CALLS   #1, STR$FREE1_DX
                                54                  50  D0 00090          MOVL    R0, STATUS
                                01              03  A2  91 00093 7$:      CMPB    3(DSC_ADDR), #1                             0368
                                                    02  12 00097          BNEQ    8$
                                                    62  B4 00099          CLRW    (DSC_ADDR)                                  0371
                                52                  53  C0 0009B 8$:      ADDL2   R3, DSC_ADDR                                0350
                                55                  52  D1 0009E 9$:      CMPL    DSC_ADDR, END_ADDR
                                                    DE  15 000A1          BLEQ    6$
                                15              02  A7  91 000A3 10$:     CMPB    2(R7), #21                                  0381
                                                    15  12 000A7          BNEQ    11$
                                51                  67  3C 000A9          MOVZWL  (R7), R1                                    0383
                51      0C      A7                  51  C7 000AC          DIVL3   R1, 12(R7), R1
                                50                  67  3C 000B1          MOVZWL  (R7), R0                                    0384
                                50                  02  C6 000B4          DIVL2   #2, R0
                                                    50  D6 000B7          INCL    R0
                                50                  51  C4 000B9          MULL2   R1, R0
                                                    04  11 000BC          BRB     12$                                         0383
                                50              0C  A7  D0 000BE 11$:     MOVL    12(R7), R0                                  0386
```

```
                                                50  D0 000C2 12$:   MOVL    R0, LENGTH                                      ; 0381
                                                58  DD 000C6       PUSHL   R8                                              ; 0387
                                            10  AE  9F 000C8       PUSHAB  LENGTH                                          ;
                                 00000000G      00  FB 000CB       CALLS   #2, LIB$FREE_VM                                 ;
                                                54  D0 000D2       MOVL    R0, STATUS                                      ;
                                                07  E8 000D5       BLBS    STATUS, 13$                                     ; 0388
                                                7E  9A 000D8       MOVZBL  #BAS$K_PROLOSSOR, -(SP)                         ; 0390
                                                6B  FB 000DC       CALLS   #1, BAS$$STOP                                   ;
                                                53  0B  A7 9A 000DF 13$:    MOVZBL  11(R7), R3                              ; 0398
                                                    52  D4 000E3       CLRL    COUNTER                                     ;
                                                    16  11 000E5       BRB     16$                                         ;
                                                04 AC42  D5 000E7 14$:    TSTL    4(AP)[COUNTER]                          ; 0400
                                                    07  18 000EB       BGEQ    15$                                         ;
                                                7E  9A 000ED       MOVZBL  #BAS$K_SUBOUTRAN, -(SP)                         ; 0402
                                                6B  FB 000F1       CALLS   #1, BAS$$STOP                                   ;
                                     50     04 AC42  C1 000F4 15$:    ADDL3   #1, 4(AP)[COUNTER], R0                  ; 0403
                                                59  C4 000FA       MULL2   R0, AR_SIZE                                     ;
                                     E6         52  F3 000FD 16$:    AOBLEQ  R3, COUNTER, 14$                            ; 0398
                                                50  3C 00101       MOVZWL  (R7), R0                                        ; 0406
                                                59  C4 00104       MULL2   R0, AR_SIZE                                     ;
                                                5A  0C  A7 9E 00107       MOVAB   12(R7), R10                             ; 0407
                                                6A  59  D0 0010B       MOVL    AR_SIZE, (R10)                              ;
                                                52  0B  A7 9A 0010E       MOVZBL  11(R7), R2                              ; 0413
                                                53  14 A742  DE 00112       MOVAL   20(R7)[R2], BOUNDS                      ;
                                                50  14  A7 9E 00117       MOVAB   20(R7), MULTIPLIERS                     ; 0414
                                     55         52  01  78 0011B       ASHL    #1, R2, R5                                  ; 0416
                                                    55  D7 0011F       DECL    R5                                          ;
                                                51      02  CE 00121       MNEGL   #2, COUNTER                             ; 0421
                                                        0B  11 00124       BRB     18$                                     ;
                                                    6341  D4 00126 17$:    CLRL    (BOUNDS)[COUNTER]                       ;
                                                04 A341  6C46  D0 00129       MOVL    (AP)[BNDSARG], 4(BOUNDS)[COUNTER]       ; 0422
                                                    56  D6 0012F       INCL    BNDSARG                                     ; 0424
                                FFEF            51      02  55  F1 00131 18$:    ACBL    R5, #2, COUNTER, 17$                ; 0416
                                                51      01  CE 00137       MNEGL   #1, COUNTER                             ; 0432
                                                        07  11 0013A       BRB     20$                                     ;
                                                6041    01  C1 0013C 19$:    ADDL3   #1, 8(AP)[COUNTER], (MULTIPLIERS)[COUNTER]
                                                  F5    52  F2 00143 20$:    AOBLSS  R2, COUNTER, 19$                    ; 0428
                                                    15  02  A7 91 00147       CMPB    2(R7), #21                          ; 0438
                                                        14  12 0014B       BNEQ    21$                                     ;
                                                51      67  3C 0014D       MOVZWL  (R7), R1                                ; 0440
                                                51      6A  51  C7 00150       DIVL3   R1, (R10), R1                       ;
                                                50      67  3C 00154       MOVZWL  (R7), R0                                ; 0441
                                                50      02  C6 00157       DIVL2   #2, R0                                  ;
                                                        50  D6 0015A       INCL    R0                                      ;
                                                50      51  C4 0015C       MULL2   R1, R0                                  ;
                                                        03  11 0015F       BRB     22$                                     ; 0440
                                                50      6A  D0 00161 21$:    MOVL    (R10), R0                           ; 0443
                                                    0C  AE  50  D0 00164 22$:    MOVL    R0, LENGTH                      ; 0438
                                                        08  AE  9F 00168       PUSHAB  EVENT_FLAG                          ; 0451
                                                 00000000G      01  FB 0016B       CALLS   #1, LIB$GET_EF                  ;
                                                                50  D0 00172       MOVL    R0, STATUS                      ;
                                                                09  E8 00175       BLBS    STATUS, 23$                     ; 0452
                                                                50  DD 00178       PUSHL   STATUS                          ;
                                                 00000000G      00  FB 0017A       CALLS   #1, LIB$STOP                    ;
                                                                7E  7C 00181 23$:    CLRQ    -(SP)                           ; 0454
                                                                7E  D4 00183       CLRL    -(SP)                           ;
                                                            1C  AE  9F 00185       PUSHAB  JPI_ITEMS                       ;
```

```
                                 7E  7C 00188          CLRQ    -(SP)                                          ;
                              20 AE  DD 0018A          PUSHL   EVENT_FLAG                                     ;
                00000000G     00  07 FB 0018D          CALLS   #7, SYS$GETJPIW                                ;
                              54  50 D0 00194          MOVL    R0, STATUS                                     ;
                              09  54 E8 00197          BLBS    STATUS, 24$                                    ; 0455
                                  54 DD 0019A          PUSHL   STATUS                                         ;
                00000000G     00  01 FB 0019C          CALLS   #1, LIB$STOP                                   ;
                              08  AE 9F 001A3 24$:      PUSHAB  EVENT_FLAG                                     ; 0457
                00000000G     00  01 FB 001A6          CALLS   #1, LIB$FREE_EF                                ;
                              54  50 D0 001AD          MOVL    R0, STATUS                                     ;
                              09  54 E8 001B0          BLBS    STATUS, 25$                                    ; 0458
                                  54 DD 001B3          PUSHL   STATUS                                         ;
                00000000G     00  01 FB 001B5          CALLS   #1, LIB$STOP                                   ;
                           50 6E  09 78 001BC 25$:      ASHL    #9, NUM_FREE_PAGES, R0                         ; 0460
                        0C AE     50 D1 001C0          CMPL    R0, LENGTH                                     ;
                                  07 1E 001C4          BGEQU   26$                                            ;
                           7E  00G 8F 9A 001C6          MOVZBL  #BAS$K_MAXMEMEXC, -(SP)                        ; 0462
                           6B     01 FB 001CA          CALLS   #1, BAS$$STOP                                  ;
                                  58 DD 001CD 26$:      PUSHL   R8                                             ; 0468
                              10  AE 9F 001CF          PUSHAB  LENGTH                                         ;
                00000000G     00  02 FB 001D2          CALLS   #2, LIB$GET_VM                                 ;
                              54  50 D0 001D9          MOVL    R0, STATUS                                     ;
                              07  54 E8 001DC          BLBS    STATUS, 27$                                    ; 0469
                           7E  00G 8F 9A 001DF          MOVZBL  #BAS$K_MAXMEMEXC, -(SP)                        ; 0471
                           6B     01 FB 001E3          CALLS   #1, BAS$$STOP                                  ;
                        10 A7     68 D0 001E6 27$:      MOVL    (R8), 16(R7)                                   ; 0472
                              02  A7 9A 001EA          MOVZBL  2(R7), R0                                      ; 0478
                                  1E 13 001EE          BEQL    30$                                            ; 0481
                              06     50 91 001F0          CMPB    R0, #6                                         ;
                                     05 1F 001F3          BLSSU   28$                                            ;
                              08     50 91 001F5          CMPB    R0, #8                                         ;
                                     14 1B 001F8          BLEQU   30$                                            ;
                              0A     50 91 001FA 28$:      CMPB    R0, #10                                        ;
                                     05 1F 001FD          BLSSU   29$                                            ;
                              0B     50 91 001FF          CMPB    R0, #11                                        ;
                                     0A 1B 00202          BLEQU   30$                                            ;
                              1B     50 91 00204 29$:      CMPB    R0, #27                                        ;
                                     3C 1F 00207          BLSSU   34$                                            ;
                              1C     50 91 00209          CMPB    R0, #28                                        ;
                                     37 1A 0020C          BGTRU   34$                                            ;
                              56     6A D0 0020E 30$:      MOVL    (R10), CURR_SIZE                               ; 0485
                              59     68 D0 00211          MOVL    (R8), CURR_ADDR                                ; 0486
                        54    68     6A C1 00214 31$:      ADDL3   (R10), (R8), R4                                ; 0494
                              54     59 D1 00218          CMPL    CURR_ADDR, R4                                  ;
                                     01 19 0021B          BLSS    32$                                            ;
                                        04 0021D          RET                                                    ;
                              50     56 D0 0021E 32$:      MOVL    CURR_SIZE, R0                                  ; 0496
                     0000FFFF 8F     50 D1 00221          CMPL    R0, #65535                                     ;
                                     05 15 00228          BLEQ    33$                                            ;
                              50 FFFF 8F 3C 0022A          MOVZWL  #65535, R0                                     ;
                50    00        6E  00 2C 0022F 33$:      MOVC5   #0, (SP), #0, R0, (CURR_ADDR)                  ;
                                    69 00234                                                                    ;
                           56 FFFF0001 E6 9E 00235          MOVAB   -65535(R6), CURR_SIZE                          ; 0497
                           59 0000FFFF E9 9E 0023C          MOVAB   65535(R9), CURR_ADDR                           ; 0498
                                     CF 11 00243          BRB     31$                                            ; 0494
                              15     50 91 00245 34$:      CMPB    R0, #21                                        ; 0502
                                     32 12 00248          BNEQ    37$                                            ;
```

```
                                59        08    A7  98 0024A          CVTBL   8(R7), COUNT                                    ; 0509
                                59              59  CE 0024E          MNEGL   COUNT, COUNT
                          54    68        0C    AE  C1 00251          ADDL3   LENGTH, (R8), R4                                ; 0510
                                50              67  3C 00256          MOVZWL  (R7), R0                                        ; 0511
                                50              02  C6 00259          DIVL2   #2, R0
                                54              50  C2 0025C          SUBL2   R0, R4
                                                54  D7 0025F          DECL    END_ADDR                                        ; 0510
                                56        01    A0  9E 00261          MOVAB   1(R0), R6                                       ; 0513
                                55              68  D0 00265          MOVL    (R8), ELE_PTR
                                                0C  11 00268          BRB     36$
                00      FD80    CF              0F  59  F8 0026A 35$:  ASHP    COUNT, #15, PACKED_ZERO, #0, (R7), -            ; 0516
                                65              67      00271                 (ELE_PTR)
                                55              55  C0 00273          ADDL2   R6, ELE_PTR                                     ; 0512
                                54              54  D1 00276 36$:     CMPL    ELE_PTR, END_ADDR
                                                EF  15 00279          BLEQ    35$
                                                    04 0027B          RET                                                     ; 0478
                                18              50  91 0027C 37$:     CMPB    R0, #24                                         ; 0520
                                                23  12 0027F          BNEQ    40$
                          54    68              6A  C1 00281          ADDL3   (R10), (R8), R4                                 ; 0524
                                51              67  3C 00285          MOVZWL  (R7), R1                                        ; 0525
                          52    54              51  C3 00288          SUBL3   R1, R4, END_ADDR
                                50              68  D0 0028C          MOVL    (R8), DSC_ADDR                                  ; 0526
                                                0D  11 0028F          BRB     39$
                                60 020E0000     8F  D0 00291 38$:     MOVL    #34471936, (DSC_ADDR)                           ; 0532
                                         04     A0  D4 00298          CLRL    4(DSC_ADDR)                                     ; 0533
                                50              51  C0 0029B          ADDL2   R1, DSC_ADDR                                    ; 0526
                                52              50  D1 0029E 39$:     CMPL    DSC_ADDR, END_ADDR
                                                EE  15 002A1          BLEQ    38$
                                                    04 002A3          RET                                                     ; 0478
                                7E        00G   8F  9A 002A4 40$:     MOVZBL  #BAS$K_NOTIMP, -(SP)                            ; 0538
                                6B              01  FB 002A8          CALLS   #1, BAS$$STOP
                                                    04 002AB          RET                                                     ; 0543

; Routine Size:  684 bytes,    Routine Base:  _BAS$CODE + 0020

;  452          0544  1 !<BLF/PAGE>
```

```
;  454          0545  1 END
;  455          0546  1
;  456          0547  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                        716  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         28         0       581          00:01.1
```

```
;                                   COMMAND QUALIFIERS

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

; Size:          684 code + 32 data bytes
; Run Time:        00:14.4
; Elapsed Time:    00:30.9
; Lines/CPU Min:    2283
; Lexemes/CPU-Min: 17657
; Memory Used:  211 pages
; Compilation Complete
```