```
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    SSSSSSSS  LL          EEEEEEEEEE  EEEEEEEEEE  PPPPPPPP
BBBBBBBB      AAAAAA      SSSSSSSS    SSSSSSSS  LL          EEEEEEEEEE  EEEEEEEEEE  PPPPPPPP
BB      BB   AA      AA  SS          SS          LL          EE          EE          PP      PP
BB      BB   AA      AA  SS          SS          LL          EE          EE          PP      PP
BB      BB   AA      AA  SS          SS          LL          EE          EE          PP      PP
BB      BB   AA      AA  SS          SS          LL          EE          EE          PP      PP
BBBBBBBB     AA      AA    SSSSSS      SSSSSS    LL          EEEEEEEE    EEEEEEEE    PPPPPPPP
BBBBBBBB     AA      AA    SSSSSS      SSSSSS    LL          EEEEEEEE    EEEEEEEE    PPPPPPPP
BB      BB   AAAAAAAAAA          SS          SS  LL          EE          EE          PP
BB      BB   AAAAAAAAAA          SS          SS  LL          EE          EE          PP
BB      BB   AA      AA          SS          SS  LL          EE          EE          PP
BB      BB   AA      AA          SS          SS  LL          EE          EE          PP
BBBBBBBB     AA      AA  SSSSSSSS    SSSSSSSS    LLLLLLLLLL  EEEEEEEEEE  EEEEEEEEEE  PP
BBBBBBBB     AA      AA  SSSSSSSS    SSSSSSSS    LLLLLLLLLL  EEEEEEEEEE  EEEEEEEEEE  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 BAS$SLEEP (                       ! Sleep for a while
;     2          0002  0                 IDENT = '3-003'          ! File: BASSLEEP.B32, Edit: FM3003
;     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:  VAX-11 BASIC Miscellaneous I/O
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains the BASIC SLEEP function.
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT:  VAX-11 User Mode
;    38          0038  1 !
;    39          0039  1 ! AUTHOR: John Sauter, CREATION DATE: 19-APR-1979, REWRITTEN: 11-JUN-1980
;    40          0040  1 !         REWRITTEN by Farokh Morshed 18-NOV-81.
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 2-001 - Rewrite this routine to use $QIO instead of RMS' read-with
;    45          0045  1 !         -timeout.  The previous version was 1-005.  JBS 11-JUN-1980
;    46          0046  1 ! 2-002 - Designate this version 2-002 to keep version numbers consistent
;    47          0047  1 !         since it is the "enhancement" version of BAS$SLEEP.  JBS 12-JUN-1980
;    48          0048  1 ! 3-001 - To implement type-ahead recovery, and SYS$INPUT translation using
;    49          0049  1 !         $PARSE, This module was rewritten.  Farokh Morshed 18-NOV-81.
;    50          0050  1 ! 3-002 - Get rid of all $TRNLOG code since system serivices do that now.
;    51          0051  1 !                 FM 14-DEC-81.
;    52          0052  1 ! 3-003 - Put in the code to get event flags from LIB$GET_EF so event flag
;    53          0053  1 !         zero is left alone.  FM 29-JUN-82.
;    54          0054  1 !--
;    55          0055  1
;    56          0056  1 !<BLF/PAGE>
```

```
;    58          0057  1 !
;    59          0058  1 ! SWITCHES:
;    60          0059  1 !
;    61          0060  1
;    62          0061  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    63          0062  1
;    64          0063  1 !
;    65          0064  1 ! LINKAGES:
;    66          0065  1 !
;    67          0066  1 !       NONE
;    68          0067  1 !
;    69          0068  1 ! TABLE OF CONTENTS:
;    70          0069  1 !
;    71          0070  1
;    72          0071  1 FORWARD ROUTINE
;    73          0072  1     BAS$SLEEP : NOVALUE,                    ! Wait some seconds
;    74          0073  1     TAKE_AST : NOVALUE,                     ! Take the AST for QIO or SETIMR
;    75          0074  1     SLEEP_HANDLER;                          ! Clean up after UNWINDs
;    76          0075  1
;    77          0076  1 !
;    78          0077  1 ! INCLUDE FILES:
;    79          0078  1 !
;    80          0079  1
;    81          0080  1 REQUIRE 'RTLIN:RTLPSECT';                   ! Macros for defining psects
;    82          0175  1
;    83          0176  1 LIBRARY 'RTLSTARLE';                        ! System symbols
;    84          0177  1
;    85          0178  1 !
;    86          0179  1 ! MACROS:
;    87          0180  1 !
;    88          0181  1 !       NONE
;    89          0182  1 !
;    90          0183  1 ! EQUATED SYMBOLS:
;    91          0184  1 !
;    92          0185  1 !       NONE
;    93          0186  1 !
;    94          0187  1 ! PSECTS:
;    95          0188  1 !
;    96          0189  1 DECLARE_PSECTS (BAS);                       ! Declare psects for BAS$ facility
;    97          0190  1 !
;    98          0191  1 ! OWN STORAGE:
;    99          0192  1 !
;   100          0193  1 !       NONE
;   101          0194  1 !
;   102          0195  1 ! EXTERNAL REFERENCES:
;   103          0196  1 !
;   104          0197  1
;   105          0198  1 EXTERNAL ROUTINE
;   106          0199  1     LIB$MATCH_COND,                         ! Match condition codes
;   107          0200  1     LIB$GET_EF,                             ! Get event flag
;   108          0201  1     LIB$FREE_EF,                            ! Free event flag
;   109          0202  1     LIB$STOP;                               ! Signal a fatal error
;   110          0203  1
```

```
;  112     0204  1 GLOBAL ROUTINE BAS$SLEEP (                      ! Wait a while
;  113     0205  1         SECONDS                                 ! Seconds to wait
;  114     0206  1     ) : NOVALUE =
;  115     0207  1
;  116     0208  1 !++
;  117     0209  1 ! FUNCTIONAL DESCRIPTION:
;  118     0210  1 !
;  119     0211  1 !       Wait the specified number of seconds, or until a terminator
;  120     0212  1 !       is typed on the controlling terminal.
;  121     0213  1 !
;  122     0214  1 ! FORMAL PARAMETERS:
;  123     0215  1 !
;  124     0216  1 !       SECONDS.rl.v    How many seconds to wait.
;  125     0217  1 !
;  126     0218  1 ! IMPLICIT INPUTS:
;  127     0219  1 !
;  128     0220  1 !       NONE
;  129     0221  1 !
;  130     0222  1 ! IMPLICIT OUTPUTS:
;  131     0223  1 !
;  132     0224  1 !       NONE
;  133     0225  1 !
;  134     0226  1 ! ROUTINE VALUE:
;  135     0227  1 ! COMPLETION CODES:
;  136     0228  1 !
;  137     0229  1 !       NONE
;  138     0230  1 !
;  139     0231  1 ! SIDE EFFECTS:
;  140     0232  1 !
;  141     0233  1 !       Signals if an error is encountered.
;  142     0234  1 !
;  143     0235  1 !--
;  144     0236  1
;  145     0237  2     BEGIN
;  146     0238  2
;  147     0239  2     LOCAL
;  148     0240  2         ASSIGNED_CHAN : VOLATILE,               !Assigned channel.  Zero if no channel is assigned.
;  149     0241  2         TIMER_REQID : VOLATILE,                 !$SETIMR request ID.
;  150     0242  2         EVENT_FLAG : VOLATILE,                  !Event flag to be used instead of EF zero.
;  151     0243  2         EF_STATUS;
;  152     0244  2
;  153     0245  2 !+
;  154     0246  2 ! Arrange to clean up if an UNWIND is done.  The most likely cause of an
;  155     0247  2 ! UNWIND is a ^C from the $HIBER.
;  156     0248  2 !-
;  157     0249  2
;  158     0250  2     ENABLE
;  159     0251  2         SLEEP_HANDLER (ASSIGNED_CHAN, TIMER_REQID, EVENT_FLAG);
;  160     0252  2
;  161     0253  2 !+
;  162     0254  2 ! Get an event flag to be used from here on.  We do this so event flag zero
;  163     0255  2 ! can be used by others.  We never use this event flag for anything.
;  164     0256  2 !-
;  165     0257  2     EF_STATUS = LIB$GET_EF (EVENT_FLAG);
;  166     0258  2     IF (NOT .EF_STATUS) THEN LIB$STOP(.EF_STATUS);
;  167     0259  2
;  168     0260  2 !+
```

```
;  169          0261  2 ! Make sure there is not a $WAKE hanging around.
;  170          0262  2 !-
;  171          0263  3     BEGIN
;  172          0264  3
;  173          0265  3     LOCAL
;  174          0266  3         WAKE_STATUS,
;  175          0267  3         HIBER_STATUS;
;  176          0268  3
;  177          0269  3     WAKE_STATUS = $WAKE ();
;  178          0270  3
;  179          0271  3     IF ( NOT .WAKE_STATUS) THEN LIB$STOP (.WAKE_STATUS);
;  180          0272  3
;  181          0273  3     HIBER_STATUS = $HIBER;
;  182          0274  3
;  183          0275  3     IF ( NOT .HIBER_STATUS) THEN LIB$STOP (.HIBER_STATUS);
;  184          0276  3
;  185          0277  2     END;
;  186          0278  3     BEGIN
;  187          0279  3
;  188          0280  3     BUILTIN
;  189          0281  3         EMUL;
;  190          0282  3
;  191          0283  3     LOCAL
;  192          0284  3         SETIMR_STATUS,
;  193          0285  3         TIMBUF : VECTOR [2];                    !Translated seconds for $SETIMR.
;  194          0286  3
;  195          0287  3 !+
;  196          0288  3 ! Compute time to wake in system format
;  197          0289  3 !-
;  198          0290  3     EMUL (%REF (-10000000), SECONDS, %REF (0), TIMBUF [0]);
;  199          0291  3 !+
;  200          0292  3 ! Take an AST when that time comes.
;  201          0293  3 ! We will pick address of SECONDS to be our TIMER_REQID since this address
;  202          0294  3 ! is unique for each call.
;  203          0295  3 !-
;  204          0296  3     TIMER_REQID = SECONDS;
;  205          0297  3     SETIMR_STATUS = $SETIMR (EFN = .EVENT_FLAG, DAYTIM = TIMBUF [0], ASTADR = TAKE_AST, REQIDT = TIMER_REQID
;  206          0298  3
;  207          0299  3     IF ( NOT .SETIMR_STATUS) THEN LIB$STOP (.SETIMR_STATUS);
;  208          0300  3
;  209          0301  2     END;
;  210          0302  2 !+
;  211          0303  2 ! Stop early if a line terminator is typed.
;  212          0304  2 !-
;  213          0305  3     BEGIN
;  214          0306  3
;  215          0307  3     LOCAL
;  216          0308  3         DEVCHR : BLOCK [DIB$K_LENGTH, BYTE],
;  217          0309  3         DEVCHR_DESC : BLOCK [8, BYTE],
;  218          0310  3         GETDEV_STATUS,
;  219          0311  3         TRNNAM_DESC : BLOCK [8, BYTE];
;  220          0312  3
;  221          0313  3     TRNNAM_DESC [DSC$W_LENGTH] = %CHARCOUNT('SYS$INPUT');
;  222          0314  3     TRNNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_Z;
;  223          0315  3     TRNNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  224          0316  3     TRNNAM_DESC [DSC$A_POINTER] = UPLIT('SYS$INPUT');
;  225          0317  3 !+
```

```
;  226          0318  3 ! Do a $GETDEV on this device name.
;  227          0319  3 !-
;  228          0320  3     DEVCHR_DESC [DSC$W_LENGTH] = DIB$K_LENGTH;
;  229          0321  3     DEVCHR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_Z;
;  230          0322  3     DEVCHR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  231          0323  3     DEVCHR_DESC [DSC$A_POINTER] = DEVCHR;
;  232          0324  4     GETDEV_STATUS = $GETDEV (DEVNAM = TRNNAM_DESC, PRILEN = DEVCHR_DESC [DSC$W_LENGTH], PRIBUF = DEVCHR_DESC
;  233          0325  3     ;
;  234          0326  3
;  235          0327  4     IF (.DEVCHR [DIB$B_DEVCLASS] EQL DC$_TERM)
;  236          0328  3     THEN
;  237          0329  3 !+
;  238          0330  3 ! SYS$INPUT is a terminal.  Arrange to take an AST if a terminator is typed
;  239          0331  3 ! on it.
;  240          0332  3 !-
;  241          0333  4         BEGIN
;  242          0334  4
;  243          0335  4         LOCAL
;  244          0336  4             QIO_STATUS,
;  245          0337  4             ASSIGN_STATUS;
;  246          0338  4
;  247          0339  4         ASSIGN_STATUS = $ASSIGN (DEVNAM = TRNNAM_DESC, CHAN = ASSIGNED_CHAN);
;  248          0340  4
;  249          0341  4         IF ( NOT .ASSIGN_STATUS) THEN LIB$STOP (.ASSIGN_STATUS);
;  250          0342  4
;  251        P 0343  4         QIO_STATUS = $QIO (EFN = .EVENT_FLAG, CHAN = .ASSIGNED_CHAN, FUNC = (IO$_SETMODE OR IO$M_OUTBAND OR
;  252          0344  4             P1 = TAKE_AST, P2 = UPLIT (0, %X'2000'));   !Terminator is a CR.
;  253          0345  4
;  254          0346  4         IF ( NOT .QIO_STATUS) THEN LIB$STOP (.QIO_STATUS);
;  255          0347  4
;  256          0348  3         END;
;  257          0349  3
;  258          0350  2     END;
;  259          0351  2 !+
;  260          0352  2 ! Now wait for the $SETIMR to fire, or (if SYS$INPUT is a terminal)
;  261          0353  2 ! for a terminator to be typed.
;  262          0354  2 !-
;  263          0355  3     BEGIN
;  264          0356  3
;  265          0357  3     LOCAL
;  266          0358  3         HIBER_STATUS;
;  267          0359  3
;  268          0360  3     HIBER_STATUS = $HIBER;
;  269          0361  3
;  270          0362  3     IF ( NOT .HIBER_STATUS) THEN LIB$STOP (.HIBER_STATUS);
;  271          0363  3
;  272          0364  2     END;
;  273          0365  2 !+
;  274          0366  2 ! At this point either AST for $SETIMR or $QIO has gone off.  We don't care
;  275          0367  2 ! which, we just cancel both of them, and also deassign the channel.
;  276          0368  2 !-
;  277          0369  3     BEGIN
;  278          0370  3
;  279          0371  3     LOCAL
;  280          0372  3         DASSGN_STATUS,
;  281          0373  3         CANTIM_STATUS,
;  282          0374  3         QIO_STATUS;
```

```
;  283          0375  3
;  284          0376  3      CANTIM_STATUS = $CANTIM (REQIDT = TIMER_REQID);
;  285          0377  3
;  286          0378  3      IF ( NOT .CANTIM_STATUS) THEN LIB$STOP (.CANTIM_STATUS);
;  287          0379  3
;  288          0380  3      IF .ASSIGNED_CHAN NEQ 0
;  289          0381  3      THEN
;  290          0382  4          BEGIN
;  291        P 0383  4          QIO_STATUS = $QIO (EFN = .EVENT_FLAG, CHAN = .ASSIGNED_CHAN, FUNC = (IO$_SETMODE OR IO$M_OUTBAND OR
;  292          0384  4              P1 = 0);
;  293          0385  4
;  294          0386  4          IF ( NOT .QIO_STATUS) THEN LIB$STOP (.QIO_STATUS);
;  295          0387  4
;  296          0388  4          DASSGN_STATUS = $DASSGN (CHAN = .ASSIGNED_CHAN);
;  297          0389  4
;  298          0390  4          IF ( NOT .DASSGN_STATUS) THEN LIB$STOP (.DASSGN_STATUS);
;  299          0391  4
;  300          0392  3          END;
;  301          0393  3
;  302          0394  2      END;
;  303          0395  2  !+
;  304          0396  2  ! Make sure there are not any $WAKE hanging around.  They could have appeared
;  305          0397  2  ! as a result of one of the ASTs timer or QIO going off just before we turned
;  306          0398  2  ! it off.
;  307          0399  2  !-
;  308          0400  3      BEGIN
;  309          0401  3
;  310          0402  3      LOCAL
;  311          0403  3          WAKE_STATUS,
;  312          0404  3          HIBER_STATUS;
;  313          0405  3
;  314          0406  3      WAKE_STATUS = $WAKE ();
;  315          0407  3
;  316          0408  3      IF ( NOT .WAKE_STATUS) THEN LIB$STOP (.WAKE_STATUS);
;  317          0409  3
;  318          0410  3      HIBER_STATUS = $HIBER;
;  319          0411  3
;  320          0412  3      IF ( NOT .HIBER_STATUS) THEN LIB$STOP (.HIBER_STATUS);
;  321          0413  3
;  322          0414  2      END;
;  323          0415  2  !+
;  324          0416  2  ! Free the event flag now
;  325          0417  2  !-
;  326          0418  2      EF_STATUS = LIB$FREE_EF (EVENT_FLAG);
;  327          0419  2      IF (NOT .EF_STATUS) THEN LIB$STOP (.EF_STATUS);
;  328          0420  2
;  329          0421  2      RETURN;
;  330          0422  1      END;                                    ! end of BAS$SLEEP


                                                                     .TITLE  BAS$SLEEP
                                                                     .IDENT  \3-003\

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

                00  00  00  54  55  50  4E  49  24  53  59  53  00000 P.AAA:  .ASCII  \SYS$INPUT\<0><0><0>
                                            00002000  00000000  0000C P.AAB:  .LONG   0, 8192
```

```
                                        .EXTRN  LIB$MATCH_COND, LIB$GET_EF
                                        .EXTRN  LIB$FREE_EF, LIB$STOP
                                        .EXTRN  SYS$WAKE, SYS$HIBER
                                        .EXTRN  SYS$SETIMR, SYS$GETDEV
                                        .EXTRN  SYS$ASSIGN, SYS$QIO
                                        .EXTRN  SYS$CANTIM, SYS$DASSGN

                        007C 00000      .ENTRY  BAS$SLEEP, Save R2,R3,R4,R5,R6                  ; 0204
                56 00000000G   00 9E 00002      MOVAB   SYS$QIO, R6                             ;
                55 00000000G   00 9E 00009      MOVAB   SYS$WAKE, R5                            ;
                54 00000000G   00 9E 00010      MOVAB   SYS$HIBER, R4                           ;
                53 00000000G   00 9E 00017      MOVAB   LIB$STOP, R3                            ;
                5E      FF70   CE 9E 0001E      MOVAB   -144(SP), SP                            ;
                         F4    AD 7C 00027      CLRQ    EVENT_FLAG                              ; 0237
                         FC    AD D4 00026      CLRL    ASSIGNED_CHAN                           ;
                6D      014E   CF DE 00029      MOVAL   14$, (FP)                               ;
                         F4    AD 9F 0002E      PUSHAB  EVENT_FLAG                              ; 0257
00000000G       00             01 FB 00031      CALLS   #1, LIB$GET_EF                          ;
                52             50 D0 00038      MOVL    R0, EF_STATUS                           ;
                05             52 E8 0003B      BLBS    EF_STATUS, 1$                           ; 0258
                               52 DD 0003E      PUSHL   EF_STATUS                               ;
                63             01 FB 00040      CALLS   #1, LIB$STOP                            ;
                               7E 7C 00043 1$:  CLRQ    -(SP)                                  ; 0269
                65             02 FB 00045      CALLS   #2, SYS$WAKE                            ;
                05             50 E8 00048      BLBS    WAKE_STATUS, 2$                         ; 0271
                               50 DD 0004B      PUSHL   WAKE_STATUS                             ;
                63             01 FB 0004D      CALLS   #1, LIB$STOP                            ;
                64             00 FB 00050 2$:  CALLS   #0, SYS$HIBER                           ; 0273
                05             50 E8 00053      BLBS    HIBER_STATUS, 3$                        ; 0275
                               50 DD 00056      PUSHL   HIBER_STATUS                            ;
                63             01 FB 00058      CALLS   #1, LIB$STOP                            ;
7C  AE          00      0C     AC FF676980  8F 7A 0005B 3$:  EMUL    #-10000000, SECONDS, #0, TIMBUF ; 0290
                         F8    AD       04  AC 9E 00066      MOVAB   SECONDS, TIMER_REQID            ; 0296
                               F8 AD       9F 0006B         PUSHAB  TIMER_REQID                     ; 0297
                                  0000V    CF 9F 0006E      PUSHAB  TAKE_AST                        ;
                                  EC       AD 9F 00072      PUSHAB  TIMBUF                          ;
                                  F4       AD DD 00075      PUSHL   EVENT_FLAG                      ;
                      00000000G   00          04 FB 00078   CALLS   #4, SYS$SETIMR                ;
                                  05          50 E8 0007F   BLBS    SETIMR_STATUS, 4$             ; 0299
                                              50 DD 00082   PUSHL   SETIMR_STATUS                 ;
                                  63          01 FB 00084   CALLS   #1, LIB$STOP                  ;
                                  6E 01000009 8F D0 00087 4$:  MOVL    #16777225, TRNNAM_DESC      ; 0313
                            04    AE    FF5A  CF 9E 0008E   MOVAB   P.AAA, TRNNAM_DESC+4          ; 0316
                            08    AE 01000074 8F D0 00094   MOVL    #16777332, DEVCHR_DESC      ; 0320
                            0C    AE       10 AE 9E 0009C   MOVAB   DEVCHR, DEVCHR_DESC+4       ; 0323
                                              7E 7C 000A1   CLRQ    -(SP)                       ; 0324
                                  10          AE 9F 000A3   PUSHAB  DEVCHR_DESC                 ;
                                  14          AE 9F 000A6   PUSHAB  DEVCHR_DESC                 ;
                                  10          AE 9F 000A9   PUSHAB  TRNNAM_DESC                 ;
                      00000000G   00          05 FB 000AC   CALLS   #5, SYS$GETDEV               ;
                            42    8F       14  AE 91 000B3      CMPB    DEVCHR+4, #66           ; 0327
                                              3D 12 000B8      BNEQ    6$                      ;
                                              7E 7C 000BA      CLRQ    -(SP)                   ; 0339
                                  FC          AD 9F 000BC      PUSHAB  ASSIGNED_CHAN           ;
                                  0C          AE 9F 000BF      PUSHAB  TRNNAM_DESC             ;
                      00000000G   00          04 FB 000C2      CALLS   #4, SYS$ASSIGN          ;
```

```
                05              50  E8 000C9          BLBS    ASSIGN_STATUS, 5$                       ; 0341
                                50  DD 000CC          PUSHL   ASSIGN_STATUS                           ;
                63              01  FB 000CE          CALLS   #1, LIB$STOP                            ;
                                7E  7C 000D1 5$:       CLRQ    -(SP)                                   ; 0344
                                7E  7C 000D3           CLRQ    -(SP)                                   ;
                          FF1F  CF  9F 000D5           PUSHAB  P.AAB                                   ;
                          0000V CF  9F 000D9           PUSHAB  TAKE_AST                                ;
                                7E  7C 000DD           CLRQ    -(SP)                                   ;
                                7E  D4 000DF           CLRL    -(SP)                                   ;
                7E        0C23  8F  3C 000E1           MOVZWL  #3107, -(SP)                            ;
                            FC  AD  DD 000E6           PUSHL   ASSIGNED_CHAN                           ;
                            F4  AD  DD 000E9           PUSHL   EVENT_FLAG                              ;
                66              0C  FB 000EC           CALLS   #12, SYS$QIO                            ;
                05              50  E8 000EF           BLBS    QIO_STATUS, 6$                          ; 0346
                                50  DD 000F2           PUSHL   QIO_STATUS                              ;
                63              01  FB 000F4           CALLS   #1, LIB$STOP                            ;
                64              00  FB 000F7 6$:       CALLS   #0, SYS$HIBER                           ; 0360
                05              50  E8 000FA           BLBS    HIBER_STATUS, 7$                        ; 0362
                                50  DD 000FD           PUSHL   HIBER_STATUS                            ;
                63              01  FB 000FF           CALLS   #1, LIB$STOP                            ;
                                7E  D4 00102 7$:       CLRL    -(SP)                                   ; 0376
                            F8  AD  9F 00104           PUSHAB  TIMER_REQID                             ;
00000000G       00              02  FB 00107           CALLS   #2, SYS$CANTIM                          ;
                05              50  E8 0010E           BLBS    CANTIM_STATUS, 8$                       ; 0378
                                50  DD 00111           PUSHL   CANTIM_STATUS                           ;
                63              01  FB 00113           CALLS   #1, LIB$STOP                            ;
                            FC  AD  D5 00116 8$:       TSTL    ASSIGNED_CHAN                           ; 0380
                                32  13 00119           BEQL    10$                                     ;
                                7E  7C 0011B           CLRQ    -(SP)                                   ; 0384
                                7E  7C 0011D           CLRQ    -(SP)                                   ;
                                7E  7C 0011F           CLRQ    -(SP)                                   ;
                                7E  7C 00121           CLRQ    -(SP)                                   ;
                                7E  D4 00123           CLRL    -(SP)                                   ;
                7E        0C23  8F  3C 00125           MOVZWL  #3107, -(SP)                            ;
                            FC  AD  DD 0012A           PUSHL   ASSIGNED_CHAN                           ;
                            F4  AD  DD 0012D           PUSHL   EVENT_FLAG                              ;
                66              0C  FB 00130           CALLS   #12, SYS$QIO                            ;
                05              50  E8 00133           BLBS    QIO_STATUS, 9$                          ; 0386
                                50  DD 00136           PUSHL   QIO_STATUS                              ;
                63              01  FB 00138           CALLS   #1, LIB$STOP                            ;
                            FC  AD  DD 0013B 9$:       PUSHL   ASSIGNED_CHAN                           ; 0388
00000000G       00              01  FB 0013E           CALLS   #1, SYS$DASSGN                          ;
                05              50  E8 00145           BLBS    DASSGN_STATUS, 10$                      ; 0390
                                50  DD 00148           PUSHL   DASSGN_STATUS                           ;
                63              01  FB 0014A           CALLS   #1, LIB$STOP                            ;
                                7E  7C 0014D 10$:      CLRQ    -(SP)                                   ; 0406
                65              02  FB 0014F           CALLS   #2, SYS$WAKE                            ;
                05              50  E8 00152           BLBS    WAKE_STATUS, 11$                        ; 0408
                                50  DD 00155           PUSHL   WAKE_STATUS                             ;
                63              01  FB 00157           CALLS   #1, LIB$STOP                            ;
                64              00  FB 0015A 11$:       CALLS   #0, SYS$HIBER                           ; 0410
                05              50  E8 0015D           BLBS    HIBER_STATUS, 12$                       ; 0412
                                50  DD 00160           PUSHL   HIBER_STATUS                            ;
                63              01  FB 00162           CALLS   #1, LIB$STOP                            ;
                            F4  AD  9F 00165 12$:      PUSHAB  EVENT_FLAG                              ; 0418
00000000G       00              01  FB 00168           CALLS   #1, LIB$FREE_EF                         ;
                52              50  D0 0016F           MOVL    R0, EF_STATUS                           ;
```

```
                      05              52  E8 00172          BLBS    EF_STATUS, 13$                          ; 0419
                                      52  DD 00175          PUSHL   EF_STATUS                               ;
                      63              01  FB 00177          CALLS   #1, LIB$STOP                            ;
                                          04 0017A 13$:      RET                                             ; 0422
                                    0000  00178 14$:        .WORD   Save nothing                            ; 0237
                      50        08    AC  D0 0017D          MOVL    8(AP), R0                               ;
                      50        04    A0  D0 00181          MOVL    4(R0), R0                               ;
                                F4    A0  9F 00185          PUSHAB  EVENT_FLAG                              ;
                                F8    A0  9F 00188          PUSHAB  TIMER_REQID                             ;
                                FC    A0  9F 0018B          PUSHAB  ASSIGNED_CHAN                           ;
                                      03  DD 0018E          PUSHL   #3                                      ;
                                      5E  DD 00190          PUSHL   SP                                      ;
                      7E        04    AC  7D 00192          MOVQ    4(AP), -(SP)                            ;
                0000V CF              03  FB 00196          CALLS   #3, SLEEP_HANDLER                       ;
                                          04 0019B          RET                                             ;

; Routine Size:  412 bytes,    Routine Base:  _BASSCODE + 0014

;  331          0423  1
```

```
;  333          0424  1 ROUTINE TAKE_AST (                            ! Take a QIO, or SETIMR AST.
;  334          0425  1         AST_PARAM                              ! AST parameter.
;  335          0426  1     ) : NOVALUE =
;  336          0427  1
;  337          0428  1 !++
;  338          0429  1 ! FUNCTIONAL DESCRIPTION:
;  339          0430  1 !
;  340          0431  1 !       Take an AST, either from $SETIMR when the sleep time is up, or from
;  341          0432  1 !       the $QIO when it completes.  In both cases we simply do a $WAKE.
;  342          0433  1 !
;  343          0434  1 ! FORMAL PARAMETERS:
;  344          0435  1 !
;  345          0436  1 !       AST_PARAM       Pointer to parameters for this AST.
;  346          0437  1 !
;  347          0438  1 ! IMPLICIT INPUTS:
;  348          0439  1 !
;  349          0440  1 !       NONE
;  350          0441  1 !
;  351          0442  1 ! IMPLICIT OUTPUTS:
;  352          0443  1 !
;  353          0444  1 !       NONE
;  354          0445  1 !
;  355          0446  1 ! COMPLETION CODES:
;  356          0447  1 !
;  357          0448  1 !       NONE
;  358          0449  1 !
;  359          0450  1 ! SIDE EFFECTS:
;  360          0451  1 !
;  361          0452  1 !       NONE
;  362          0453  1 !
;  363          0454  1 !--
;  364          0455  1
;  365          0456  2     BEGIN
;  366          0457  2     $WAKE ();
;  367          0458  2     RETURN;
;  368          0459  1     END;                                       ! of TAKE_AST


                                0000 00000 TAKE_AST:
                                           .WORD   Save nothing                    ; 0424
                             7E  7C 00002          CLRQ    -(SP)                   ; 0457
                00000000G    00 02  FB 00004       CALLS   #2, SYS$WAKE            ;
                                04 0000B           RET                             ; 0459

; Routine Size:  12 bytes,    Routine Base:  _BAS$CODE + 01B0
```

```
;  370          0460  1 ROUTINE SLEEP_HANDLER (                       ! Handler for BAS$SLEEP
;  371          0461  1         SIG,                                   ! signal args
;  372          0462  1         MECH,                                  ! mechanism args
;  373          0463  1         ENBL                                   !variables passed from BAS$SLEEP
;  374          0464  1     ) =
;  375          0465  1
;  376          0466  1 !++
;  377          0467  1 ! FUNCTIONAL DESCRIPTION:
;  378          0468  1 !
;  379          0469  1 !       Handle an UNWIND while in BAS$SLEEP.  This is needed so that the
;  380          0470  1 !       ASTs will not fire after their storage has been removed from the
;  381          0471  1 !       stack.
;  382          0472  1 !
;  383          0473  1 ! FORMAL PARAMETERS:
;  384          0474  1 !
;  385          0475  1 !       SIG.rl.a        Address of the signal vector.  This contains
;  386          0476  1 !                       the condition.
;  387          0477  1 !       MECH.rl.a       Address of the mechanism vector.  This contains
;  388          0478  1 !                       the status of the frame that signalled.
;  389          0479  1 !       ENBL.rl.a       Address of the enable vector.  This contains
;  390          0480  1 !                       some the local variables ASSIGNED_CHAN, and TIMER_REQID.
;  391          0481  1 !
;  392          0482  1 ! IMPLICIT INPUTS:
;  393          0483  1 !
;  394          0484  1 !       NONE
;  395          0485  1 !
;  396          0486  1 ! IMPLICIT OUTPUTS:
;  397          0487  1 !
;  398          0488  1 !       NONE
;  399          0489  1 !
;  400          0490  1 ! COMPLETION CODES:
;  401          0491  1 !
;  402          0492  1 !       Always SS$_RESIGNAL, but this is ingored when we are
;  403          0493  1 !       unwinding.
;  404          0494  1 !
;  405          0495  1 ! SIDE EFFECTS:
;  406          0496  1 !
;  407          0497  1 !       Arranges that the ASTs will not fire after this routine
;  408          0498  1 !       has completed its execution.
;  409          0499  1 !
;  410          0500  1 !--
;  411          0501  1
;  412          0502  2     BEGIN
;  413          0503  2
;  414          0504  2     MAP
;  415          0505  2         SIG : REF VECTOR,                      ! signal vector
;  416          0506  2         MECH : REF VECTOR,                     ! mechanism vector
;  417          0507  2         ENBL : REF VECTOR;                     ! enable vector
;  418          0508  2
;  419          0509  2     BIND
;  420          0510  2         ASSIGNED_CHAN = .ENBL [1],
;  421          0511  2         TIMER_REQID = .ENBL [2],
;  422          0512  2         EVENT_FLAG = .ENBL [3];
;  423          0513  2
;  424          0514  2 !+
;  425          0515  2 ! If this is the UNWIND condition, cancel the SETIMR and QIO.
;  426          0516  2 !-
```

```
;  427          0517  2
;  428          0518  3      IF (LIB$MATCH_COND (SIG [1], %REF (SS$_UNWIND)))
;  429          0519  2      THEN
;  430          0520  3          BEGIN
;  431          0521  3  !+
;  432          0522  3  ! Turn off the QIO and SETIMR.  We need to do this while no ASTs can go off
;  433          0523  3  ! because we are modifying ASSIGNED_CHAN, and TIMER_REQID.
;  434          0524  3  !-
;  435          0525  3          $SETAST (ENBFLG = 0);
;  436          0526  4          BEGIN
;  437          0527  4
;  438          0528  4          LOCAL
;  439          0529  4              DASSGN_STATUS,
;  440          0530  4              CANTIM_STATUS,
;  441          0531  4              EF_STATUS,
;  442          0532  4              QIO_STATUS;
;  443          0533  4
;  444        P 0534  4          QIO_STATUS = $QIO (EFN = .EVENT_FLAG, CHAN = .ASSIGNED_CHAN, FUNC = (IO$_SETMODE OR IO$M_OUTBAND OR
;  445          0535  4              P1 = 0);
;  446          0536  4
;  447          0537  4          IF ( NOT .QIO_STATUS) THEN LIB$STOP (.QIO_STATUS);
;  448          0538  4
;  449          0539  4          EF_STATUS = LIB$FREE_EF (EVENT_FLAG);
;  450          0540  4
;  451          0541  4          IF .ASSIGNED_CHAN NEQ 0
;  452          0542  4          THEN
;  453          0543  5              BEGIN
;  454          0544  5              DASSGN_STATUS = $DASSGN (CHAN = .ASSIGNED_CHAN);
;  455          0545  5
;  456          0546  5              IF ( NOT .DASSGN_STATUS) THEN LIB$STOP (.DASSGN_STATUS);
;  457          0547  5
;  458          0548  5              ASSIGNED_CHAN = 0;
;  459          0549  4              END;
;  460          0550  4
;  461          0551  4          IF .TIMER_REQID NEQ 0
;  462          0552  4          THEN
;  463          0553  5              BEGIN
;  464          0554  5              CANTIM_STATUS = $CANTIM (REQIDT = TIMER_REQID);
;  465          0555  5
;  466          0556  5              IF ( NOT .CANTIM_STATUS) THEN LIB$STOP (.CANTIM_STATUS);
;  467          0557  5
;  468          0558  5              TIMER_REQID = 0;
;  469          0559  4              END;
;  470          0560  4
;  471          0561  3          END;
;  472          0562  3          $SETAST (ENBFLG = 1);
;  473          0563  3  !+
;  474          0564  3  ! Make sure there are not any $WAKE hanging around.  They could have appeared
;  475          0565  3  ! as a result of one of the ASTs timer or QIO going off just before we turned
;  476          0566  3  ! it off.
;  477          0567  3  !-
;  478          0568  4          BEGIN
;  479          0569  4
;  480          0570  4          LOCAL
;  481          0571  4              WAKE_STATUS,
;  482          0572  4              HIBER_STATUS;
;  483          0573  4
```

```
;  484          0574  4          WAKE_STATUS = $WAKE ();
;  485          0575  4
;  486          0576  4          IF ( NOT .WAKE_STATUS) THEN LIB$STOP (.WAKE_STATUS);
;  487          0577  4
;  488          0578  4          HIBER_STATUS = $HIBER;
;  489          0579  4
;  490          0580  4          IF ( NOT .HIBER_STATUS) THEN LIB$STOP (.HIBER_STATUS);
;  491          0581  4
;  492          0582  3          END;
;  493          0583  2          END:
;  494          0584  2
;  495          0585  2      RETURN (SS$_RESIGNAL);
;  496          0586  1      END;                                    ! of HANDLER


                                                        .EXTRN  SYS$SETAST

                                        001C 00000 SLEEP_HANDLER:
                                                        .WORD   Save R2,R3,R4                   ; 0460
                          54 00000000G   00   9E 00002          MOVAB   SYS$SETAST, R4          ;
                          53 00000000G   00   9E 00009          MOVAB   LIB$STOP, R3            ;
                          52       0C    AC   D0 00010          MOVL    ENBL, R2                ; 0510
                          7E     0920    8F   3C 00014          MOVZWL  #2336, -(SP)            ; 0518
                                         5E   DD 00019          PUSHL   SP                      ;
                    7E          04    AC        04   C1 0001B    ADDL3   #4, SIG, -(SP)          ;
                 00000000G      00              02   FB 00020    CALLS   #2, LIB$MATCH_COND      ;
                                03              50   E8 00027    BLBS    R0, 1$                  ;
                                             008E   31 0002A    BRW     8$                      ;
                                               7E   D4 0002D 1$: CLRL    -(SP)                   ; 0525
                                64             01   FB 0002F    CALLS   #1, SYS$SETAST          ;
                                               7E   7C 00032    CLRQ    -(SP)                   ; 0535
                                               7E   7C 00034    CLRQ    -(SP)                   ;
                                               7E   7C 00036    CLRQ    -(SP)                   ;
                                               7E   7C 00038    CLRQ    -(SP)                   ;
                                               7E   D4 0003A    CLRL    -(SP)                   ;
                                            7E      0C23   8F   3C 0003C    MOVZWL  #3107, -(SP)    ;
                                                     04   B2   DD 00041    PUSHL   @4(R2)          ;
                                                     0C   B2   DD 00044    PUSHL   @12(R2)         ;
                                00000000G   00            0C   FB 00047    CALLS   #12, SYS$QIO    ;
                                            05            50   E8 0004E    BLBS    QIO_STATUS, 2$  ; 0537
                                                          50   DD 00051    PUSHL   QIO_STATUS      ;
                                            63            01   FB 00053    CALLS   #1, LIB$STOP    ;
                                                     0C   A2   DD 00056 2$: PUSHL   12(R2)         ; 0539
                                00000000G   00            01   FB 00059    CALLS   #1, LIB$FREE_EF ;
                                                     04   B2   D5 00060    TSTL    @4(R2)          ; 0541
                                                          15   13 00063    BEQL    4$              ;
                                                     04   B2   DD 00065    PUSHL   @4(R2)          ; 0544
                                00000000G   00            01   FB 00068    CALLS   #1, SYS$DASSGN  ;
                                            05            50   E8 0006F    BLBS    DASSGN_STATUS, 3$ ; 0546
                                                          50   DD 00072    PUSHL   DASSGN_STATUS   ;
                                            63            01   FB 00074    CALLS   #1, LIB$STOP    ;
                                                     04   B2   D4 00077 3$: CLRL    @4(R2)          ; 0548
                                                     08   B2   D5 0007A 4$: TSTL    @8(R2)          ; 0551
                                                          17   13 0007D    BEQL    6$              ;
                                                          7E   D4 0007F    CLRL    -(SP)           ; 0554
                                                     08   A2   DD 00081    PUSHL   8(R2)           ;
```

```
                    00000000G  00          02  FB 00084          CALLS   #2, SYS$CANTIM                          ;
                               05          50  E8 0008B          BLBS    CANTIM_STATUS, 5$                      ; 0556
                                           50  DD 0008E          PUSHL   CANTIM_STATUS                          ;
                               63          01  FB 00090          CALLS   #1, LIB$STOP                           ;
                                     08    B2  D4 00093 5$:       CLRL    @8(R2)                                 ; 0558
                                           01  DD 00096 6$:       PUSHL   #1                                     ; 0562
                               64          01  FB 00098          CALLS   #1, SYS$SETAST                         ;
                                           7E  7C 0009B          CLRQ    -(SP)                                  ; 0574
                    00000000G  00          02  FB 0009D          CALLS   #2, SYS$WAKE                           ;
                               05          50  E8 000A4          BLBS    WAKE_STATUS, 7$                        ; 0576
                                           50  DD 000A7          PUSHL   WAKE_STATUS                            ;
                               63          01  FB 000A9          CALLS   #1, LIB$STOP                           ;
                    00000000G  00          00  FB 000AC 7$:      CALLS   #0, SYS$HIBER                          ; 0578
                               05          50  E8 000B3          BLBS    HIBER_STATUS, 8$                       ; 0580
                                           50  DD 000B6          PUSHL   HIBER_STATUS                           ;
                               63          01  FB 000B8          CALLS   #1, LIB$STOP                           ;
                               50    0918  8F  3C 000BB 8$:      MOVZWL  #2328, R0                              ; 0585
                                           04  000C0            RET                                            ; 0586

; Routine Size:  193 bytes,    Routine Base:  _BAS$CODE + 01BC

;  497          0587  1 END                                    ! end of module BAS$SLEEP
;  498          0588  1
;  499          0589  0 ELUDOM
```

```
;                                PSECT SUMMARY

;     Name                      Bytes                          Attributes

; _BAS$CODE                        637  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        25         0       581         00:01.1


;                                  COMMAND QUALIFIERS

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

```
; Size:          617 code + 20 data bytes
; Run Time:        00:12.7
; Elapsed Time:    00:29.9
; Lines/CPU Min:    2784
; Lexemes/CPU-Min: 21976
; Memory Used:  143 pages
; Compilation Complete
```