```
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
```

FILEID**BASDATETI

```
BBBBBBBB      AAAAAA      SSSSSSSS  DDDDDDDD      AAAAAA   TTTTTTTTTT  EEEEEEEEEE  TTTTTTTTTT    IIIIII
BBBBBBBB      AAAAAA      SSSSSSSS  DDDDDDDD      AAAAAA   TTTTTTTTTT  EEEEEEEEEE  TTTTTTTTTT    IIIIII
BB      BB   AA      AA  SS        DD      DD   AA      AA      TT      EE              TT        II
BB      BB   AA      AA  SS        DD      DD   AA      AA      TT      EE              TT        II
BB      BB   AA      AA  SS        DD      DD   AA      AA      TT      EE              TT        II
BB      BB   AA      AA  SS        DD      DD   AA      AA      TT      EE              TT        II
BBBBBBBB     AA      AA    SSSSSS  DD      DD   AA      AA      TT      EEEEEEEE        TT        II
BBBBBBBB     AA      AA    SSSSSS  DD      DD   AA      AA      TT      EEEEEEEE        TT        II
BB      BB   AAAAAAAAAA          SS  DD      DD   AAAAAAAAAA      TT      EE              TT        II
BB      BB   AAAAAAAAAA          SS  DD      DD   AAAAAAAAAA      TT      EE              TT        II
BB      BB   AA      AA          SS  DD      DD   AA      AA      TT      EE              TT        II          ....
BB      BB   AA      AA          SS  DD      DD   AA      AA      TT      EE              TT        II          ....
BBBBBBBB     AA      AA  SSSSSSSS    DDDDDDDD     AA      AA      TT      EEEEEEEEEE      TT      IIIIII        ....
BBBBBBBB     AA      AA  SSSSSSSS    DDDDDDDD     AA      AA      TT      EEEEEEEEEE      TT      IIIIII        ....

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$DATE_TIME (                      ! Date and Time functions
;     2          0002  0                 IDENT = '1-015'            ! File: BASDATETI.B32 EDIT:STAN1015
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**********************************************************************
;     7          0007  1 !*                                                                    *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                          *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.           *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                           *
;    11          0011  1 !*                                                                    *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                       *
;    18          0018  1 !*                                                                    *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                       *
;    22          0022  1 !*                                                                    *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    25          0025  1 !*                                                                    *
;    26          0026  1 !*                                                                    *
;    27          0027  1 !**********************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:  BASIC-PLUS-2 Miscellaneous Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains the DATE and TIME functions as defined
;    36          0036  1 !       in the BASIC-PLUS-2 language manual.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  VAX-11 User Mode
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: John Sauter, CREATION DATE: 19-FEB-1979
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original.  JBS 19-FEB-1979
;    45          0045  1 ! 1-002 - Correct the computation of DATE$ to give a positive absolute
;    46          0046  1 !           time for the last call to $ASCTIM.  JBS 23-FEB-1979
;    47          0047  1 ! 1-003 - Correct bad offsets for the current time field in the string
;    48          0048  1 !           returned by LIB$DATE_TIME.  This will have to be tested at
;    49          0049  1 !           several hours of the day to verify that the offsets are
;    50          0050  1 !           fixed.  JBS 07-MAR-1979
;    51          0051  1 ! 1-004 - Return the second and third letters of the month in lower
;    52          0052  1 !           case.  JBS 07-MAR-1979
;    53          0053  1 ! 1-005 - Change the string entry point names to end with _T rather than
;    54          0054  1 !           DX.  JBS 19-MAR-1979
;    55          0055  1 ! 1-006 - Change LIB$S and OTS$S to STR$.  JBS 21-MAY-1979
;    56          0056  1 ! 1-007 - Change calls to STR$COPY.  JBS 16-JUL-1979
;    57          0057  1 ! 1-008 - Correct a comment.  JBS 07-NOV-1979
```

```
;    58          0058  1 ! 1-009 - Add bounds checking for TIME$.  PLL 11-MAY-1981
;    59          0059  1 ! 1-010 - DATE$ should return 00-XXX-00 when passed an invalid
;    60          0060  1 !         argument.  PL 05-JUN-81
;    61          0061  1 ! 1-011 - LIB$STOP should be declared EXTERNAL.  PLL 20-Nov-81
;    62          0062  1 ! 1-012 - Use LIB$GET_EF to obtain an event flag for $GETJPI.  If none is
;    63          0063  1 !         specified, efn zero is used, and this may interfere with other
;    64          0064  1 !         procedures.  PLL 30-Nov-81
;    65          0065  1 ! 1-013 - Finish last edit.  PLL 1-Dec-81
;    66          0066  1 ! 1-014 - Performance improvements on TIME(0), TIME$(0), DATE$(0).  Now
;    67          0067  1 !         calls $ASCTIM instead of LIB$DATE_TIME.  MDL 12-Jul-1982
;    68          0068  1 ! 1-015 - Remove informational errors. STAN 24-Jul-1984.
;    69          0069  1 !
;    70          0070  1 !--
;    71          0071  1
;    72          0072  1 !<BLF/PAGE>
```

```
;    74          0073  1 !
;    75          0074  1 ! SWITCHES:
;    76          0075  1 !
;    77          0076  1
;    78          0077  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    79          0078  1
;    80          0079  1 !
;    81          0080  1 ! LINKAGES:
;    82          0081  1 !
;    83          0082  1 !       NONE
;    84          0083  1 !
;    85          0084  1 ! TABLE OF CONTENTS:
;    86          0085  1 !
;    87          0086  1
;    88          0087  1 FORWARD ROUTINE
;    89          0088  1     BAS$DATE_T : NOVALUE,                   ! Perform a DATE$ function
;    90          0089  1     BAS$TIME_T : NOVALUE,                   ! Perform a TIME$ function
;    91          0090  1     BAS$TIME_F;                             ! Perform a TIME function
;    92          0091  1
;    93          0092  1 !
;    94          0093  1 ! INCLUDE FILES:
;    95          0094  1 !
;    96          0095  1
;    97          0096  1 REQUIRE 'RTLIN:RTLPSECT';                   ! Macros for defining psects
;    98          0191  1
;    99          0192  1 LIBRARY 'RTLSTARLE';                        ! Define system symbols
;   100          0193  1
;   101          0194  1 !
;   102          0195  1 ! MACROS:
;   103          0196  1 !
;   104          0197  1 !       NONE
;   105          0198  1 !
;   106          0199  1 ! EQUATED SYMBOLS:
;   107          0200  1 !
;   108          0201  1 !       NONE
;   109          0202  1 !
;   110          0203  1 ! PSECTS:
;   111          0204  1 !
;   112          0205  1 DECLARE_PSECTS (BAS);                       ! Declare psects for BAS$ facility
;   113          0206  1 !
;   114          0207  1 ! OWN STORAGE:
;   115          0208  1 !
;   116          0209  1 !       NONE
;   117          0210  1 !
;   118          0211  1 ! EXTERNAL REFERENCES:
;   119          0212  1 !
;   120          0213  1
;   121          0214  1 EXTERNAL ROUTINE
;   122          0215  1     LIB$GET_EF,                             ! allocate an event flag
;   123          0216  1     LIB$FREE_EF,                            ! deallocate an event flag
;   124          0217  1     LIB$STOP : NOVALUE,                     ! signal fatal error
;   125          0218  1     LIB$SUBX,                               ! Subtract 64-bit integers
;   126          0219  1     STR$COPY_R,                             ! Copy a string by reference
;   127          0220  1     BAS$$STOP : NOVALUE,                    ! signals fatal error
;   128          0221  1     BAS$$SIGNAL : NOVALUE;                  ! signals an error
;   129          0222  1
;   130          0223  1 !+
```

```
;  131          0224  1 ! The following are the error codes used in this module.
;  132          0225  1 !-
;  133          0226  1
;  134          0227  1 EXTERNAL LITERAL
;  135          0228  1     BAS$K_ARGOUTBOU : UNSIGNED (8),          ! Argument Out of Bounds
;  136          0229  1     BAS$K_NOTIMP : UNSIGNED (8),             ! Not implemented
;  137          0230  1     OTS$_FATINTERR;                          ! OTS Fatal Internal Error
;  138          0231  1
```

```
;  140     0232  1 GLOBAL ROUTINE BAS$DATE_T (                   ! Perform a DATE$ function
;  141     0233  1         DATE_STR,                              ! Resulting string
;  142     0234  1         DAYNO                                  ! The day number, as defined below
;  143     0235  1     ) : NOVALUE =
;  144     0236  1
;  145     0237  1 !++
;  146     0238  1 ! FUNCTIONAL DESCRIPTION:
;  147     0239  1 !
;  148     0240  1 !       Perform a DATE$ function, as follows:
;  149     0241  1 !
;  150     0242  1 !       DATE$(0%) returns the current date in the form dd-Mmm-yy
;  151     0243  1 !       DATE$(n%) returns the date corresponding to day number
;  152     0244  1 !               n, where n is the day of the year (1 to 365 or 366)
;  153     0245  1 !               plus the number of years since 00-Jan-1970 * 1000.
;  154     0246  1 !
;  155     0247  1 ! FORMAL PARAMETERS:
;  156     0248  1 !
;  157     0249  1 !       DATE_STR.wt.d   The result string
;  158     0250  1 !       DAYNO.rl.v      The day number, or zero.
;  159     0251  1 !
;  160     0252  1 ! IMPLICIT INPUTS:
;  161     0253  1 !
;  162     0254  1 !       The system date and time, if DAYNO = 0.
;  163     0255  1 !
;  164     0256  1 ! IMPLICIT OUTPUTS:
;  165     0257  1 !
;  166     0258  1 !       NONE
;  167     0259  1 !
;  168     0260  1 ! ROUTINE VALUE:
;  169     0261  1 ! COMPLETION CODES:
;  170     0262  1 !
;  171     0263  1 !       NONE
;  172     0264  1 !
;  173     0265  1 ! SIDE EFFECTS:
;  174     0266  1 !
;  175     0267  1 !       NONE
;  176     0268  1 !
;  177     0269  1 !--
;  178     0270  1
;  179     0271  2     BEGIN
;  180     0272  2 !+
;  181     0273  2 ! If the day number is zero, get the system day and time, and reformat
;  182     0274  2 ! it to conform to BASIC-PLUS-2's standard.
;  183     0275  2 !-
;  184     0276  2
;  185     0277  3     IF (.DAYNO EQL 0)
;  186     0278  2     THEN
;  187     0279  3         BEGIN
;  188     0280  3
;  189     0281  3         LOCAL
;  190     0282  3             DATE_DESC : BLOCK [8, BYTE],
;  191     0283  3             DATE_BUF : VECTOR [24, BYTE];
;  192     0284  3
;  193     0285  3 !+
;  194     0286  3 ! Build the descriptor for the date string which will contain today's date.
;  195     0287  3 !-
;  196     0288  3         DATE_DESC [DSC$W_LENGTH] = 11;
```

```
;  197     0289  3          DATE_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  198     0290  3          DATE_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  199     0291  3          DATE_DESC [DSC$A_POINTER] = DATE_BUF;
;  200     0292  3
;  201     0293  3      !+
;  202     0294  3      ! get the current date
;  203     0295  3      !-
;  204     0296  3          $ASCTIM ( TIMBUF = DATE_DESC );
;  205     0297  3
;  206     0298  3      !+
;  207     0299  3      ! Suppress the century by putting the year on top of it.
;  208     0300  3      !-
;  209     0301  3          DATE_BUF [7] = .DATE_BUF [9];
;  210     0302  3          DATE_BUF [8] = .DATE_BUF [10];
;  211     0303  3      !+
;  212     0304  3      ! Make sure the leading character of the day is a zero rather rhan a blank.
;  213     0305  3      !-
;  214     0306  3
;  215     0307  3          IF (.DATE_BUF [0] EQL %C' ') THEN DATE_BUF [0] = %C'0';
;  216     0308  3
;  217     0309  3      !+
;  218     0310  3      ! Make sure that the second and third characters of the month name are
;  219     0311  3      ! in lower case.
;  220     0312  3      !-
;  221     0313  3          DATE_BUF [4] = .DATE_BUF [4] OR 32;
;  222     0314  3          DATE_BUF [5] = .DATE_BUF [5] OR 32;
;  223     0315  3      !+
;  224     0316  3      ! Return the string to the user.  We return only the first nine characters.
;  225     0317  3      !-
;  226     0318  3          STR$COPY_R (.DATE_STR, %REF (9), DATE_BUF [0]);
;  227     0319  3          RETURN;
;  228     0320  2          END;
;  229     0321  2
;  230     0322  2      !+
;  231     0323  2      ! The day number is not zero.  Compute a date based on it.  The year is
;  232     0324  2      ! the day number modulo 1000, the day of the year is the day number
;  233     0325  2      ! divided by 1000.  To avoid having tables of the number of days in each
;  234     0326  2      ! month, and to avoid worrying about leap year (is the year 2100 a leap
;  235     0327  2      ! year?) we use the system time services.
;  236     0328  2      !-
;  237     0329  3          BEGIN
;  238     0330  3
;  239     0331  3          LOCAL
;  240     0332  3              DATE_BUF : VECTOR [24, BYTE],
;  241     0333  3              DAY_BUF : VECTOR [4, BYTE],
;  242     0334  3              DATE_DESC : BLOCK [8, BYTE],
;  243     0335  3              DAY_DESC : BLOCK [8, BYTE],
;  244     0336  3              YEAR,
;  245     0337  3              DAY,
;  246     0338  3              Q_BASE_DATE : VECTOR [2],
;  247     0339  3              Q_DELTA_DAYS : VECTOR [2],
;  248     0340  3              Q_FINAL_DATE : VECTOR [2];
;  249     0341  3
;  250     0342  3      !+
;  251     0343  3      ! Set up a dummy date string in case the DATE$ argument is invalid.
;  252     0344  3      !-
;  253     0345  3          DATE_BUF [0] = %C'0';
```

```
;  254          0346  3          DATE_BUF [1] = %C'0';
;  255          0347  3          DATE_BUF [2] = %C'-';
;  256          0348  3          DATE_BUF [3] = %C'X';
;  257          0349  3          DATE_BUF [4] = %C'X';
;  258          0350  3          DATE_BUF [5] = %C'X';
;  259          0351  3          DATE_BUF [6] = %C'-';
;  260          0352  3          DATE_BUF [7] = %C'0';
;  261          0353  3          DATE_BUF [8] = %C'0';
;  262          0354  3      !+
;  263          0355  3      ! If the argument is a negative number or the day is greater than 366, it's
;  264          0356  3      ! definitely invalid.  Return 00-XXX-00.  If the day number is 366,
;  265          0357  3      ! then make sure that year was a leap year.
;  266          0358  3      !-
;  267          0359  3
;  268          0360  3          DAY = (.DAYNO) MOD 1000;
;  269          0361  3          YEAR = 1970 + (.DAYNO/1000);
;  270          0362  4          IF (.DAYNO LSS 0) OR (.DAY GTR 366)
;  271          0363  3          THEN
;  272          0364  4              BEGIN
;  273          0365  4              STR$COPY_R (.DATE_STR, %REF(9), DATE_BUF [0]);
;  274          0366  4              RETURN;
;  275          0367  3              END;
;  276          0368  4          IF (.DAY EQL 366)
;  277          0369  3          THEN
;  278          0370  4              BEGIN
;  279          0371  5              IF ((.YEAR MOD 4) NEQ 0)
;  280          0372  4              THEN
;  281          0373  5                  BEGIN
;  282          0374  5                  STR$COPY_R (.DATE_STR, %REF (9), DATE_BUF [0]);
;  283          0375  5                  RETURN;
;  284          0376  4                  END;
;  285          0377  3              END;
;  286          0378  3
;  287          0379  3      !+
;  288          0380  3      ! Compute the binary time corresponding to the base date, which is
;  289          0381  3      ! 00-JAN-1970 plus the day number modulo 1000.
;  290          0382  3      !-
;  291          0383  3          DATE_BUF [0] = %C'0';
;  292          0384  3          DATE_BUF [1] = %C'1';
;  293          0385  3          DATE_BUF [2] = %C'-';
;  294          0386  3          DATE_BUF [3] = %C'J';
;  295          0387  3          DATE_BUF [4] = %C'A';
;  296          0388  3          DATE_BUF [5] = %C'N';
;  297          0389  3          DATE_BUF [6] = %C'-';
;  298          0390  3          DATE_BUF [7] = ((.YEAR/1000) + %C'0');
;  299          0391  3          DATE_BUF [8] = (((.YEAR/100) MOD 10) + %C'0');
;  300          0392  3          DATE_BUF [9] = (((.YEAR/10) MOD 10) + %C'0');
;  301          0393  3          DATE_BUF [10] = ((.YEAR MOD 10) + %C'0');
;  302          0394  3          DATE_BUF [11] = %C' ';
;  303          0395  3          DATE_BUF [12] = %C'0';
;  304          0396  3          DATE_BUF [13] = %C'0';
;  305          0397  3          DATE_BUF [14] = %C':';
;  306          0398  3          DATE_BUF [15] = %C'0';
;  307          0399  3          DATE_BUF [16] = %C'0';
;  308          0400  3          DATE_BUF [17] = %C':';
;  309          0401  3          DATE_BUF [18] = %C'0';
;  310          0402  3          DATE_BUF [19] = %C'0';
```

```
;  311          0403  3      DATE_BUF [20] = %C',';
;  312          0404  3      DATE_BUF [21] = %C'0';
;  313          0405  3      DATE_BUF [22] = %C'0';
;  314          0406  3      DATE_BUF [23] = %C' ';
;  315          0407  3  !+
;  316          0408  3  ! Convert that to absolute system time format, which is a 64-bit integer.
;  317          0409  3  !-
;  318          0410  3      DATE_DESC [DSC$W_LENGTH] = 24;
;  319          0411  3      DATE_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  320          0412  3      DATE_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  321          0413  3      DATE_DESC [DSC$A_POINTER] = DATE_BUF [0];
;  322          0414  3      $BINTIM (TIMBUF = DATE_DESC, TIMADR = Q_BASE_DATE);
;  323          0415  3  !+
;  324          0416  3  !+
;  325          0417  3  ! Now convert the specified number of days into a system-format
;  326          0418  3  ! delta time.
;  327          0419  3  !-
;  328          0420  3      DAY_BUF [0] = %C'0';
;  329          0421  3      DAY_BUF [1] = (((.DAYNO - 1)/100) MOD 10) + %C'0';
;  330          0422  3      DAY_BUF [2] = (((.DAYNO - 1)/10) MOD 10) + %C'0';
;  331          0423  3      DAY_BUF [3] = ((.DAYNO - 1) MOD 10) + %C'0';
;  332          0424  3      DAY_DESC [DSC$W_LENGTH] = 4;
;  333          0425  3      DAY_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  334          0426  3      DAY_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  335          0427  3      DAY_DESC [DSC$A_POINTER] = DAY_BUF [0];
;  336          0428  3      $BINTIM (TIMBUF = DAY_DESC, TIMADR = Q_DELTA_DAYS);
;  337          0429  3  !+
;  338          0430  3  ! Add the delta time to the base time.  This must be done with a subtract
;  339          0431  3  ! since the delta time is kept in negative format.  Also, it must be done
;  340          0432  3  ! with quadword arithmetic.
;  341          0433  3  !-
;  342          0434  3      LIB$SUBX (Q_BASE_DATE, Q_DELTA_DAYS, Q_FINAL_DATE, %REF (2));
;  343          0435  3  !+
;  344          0436  3  ! Now reconvert the system date to a readable date.
;  345          0437  3  !-
;  346          0438  3      $ASCTIM (TIMBUF = DATE_DESC, TIMADR = Q_FINAL_DATE);
;  347          0439  3  !+
;  348          0440  3  ! Proceed as above to suppress the century and return the nine-character
;  349          0441  3  ! string dd-Mmm-yy
;  350          0442  3  !-
;  351          0443  3      DATE_BUF [7] = .DATE_BUF [9];
;  352          0444  3      DATE_BUF [8] = .DATE_BUF [10];
;  353          0445  3
;  354          0446  3      IF (.DATE_BUF [0] EQL %C' ') THEN DATE_BUF [0] = %C'0';
;  355          0447  3
;  356          0448  3      DATE_BUF [4] = .DATE_BUF [4] OR 32;
;  357          0449  3      DATE_BUF [5] = .DATE_BUF [5] OR 32;
;  358          0450  3      STR$COPY_R (.DATE_STR, %REF (9), DATE_BUF [0]);
;  359          0451  3      RETURN;
;  360          0452  2      END;
;  361          0453  1      END;                                          ! end of BAS$DATE_T

                                                                           .TITLE  BAS$DATE_TIME
                                                                           .IDENT  \1-015\

                                                                           .EXTRN  LIB$GET_EF, LIB$FREE_EF
```

```
                                        .EXTRN  LIB$STOP, LIB$SUBX
                                        .EXTRN  STR$COPY_R, BAS$$STOP
                                        .EXTRN  BAS$$SIGNAL, BAS$K_ARGOUTBOU
                                        .EXTRN  BAS$K_NOTIMP, OTS$_FATINTERR
                                        .EXTRN  SYS$ASCTIM, SYS$BINTIM

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

                                007C 00000      .ENTRY  BAS$DATE_T, Save R2,R3,R4,R5,R6                 ; 0232
                    56 00000000G   00   9E 00002        MOVAB   SYS$ASCTIM, R6
                    55 00000000G   00   9E 00009        MOVAB   SYS$BINTIM, R5
                    5E        B8   AE   9E 00010        MOVAB   -72(SP), SP
                    54        08   AC   D0 00014        MOVL    DAYNO, R4                               ; 0277
                                   32   12 00018        BNEQ    2$
                40  AE 010E000B   8F   D0 0001A        MOVL    #17694731, DATE_DESC                    ; 0288
                44  AE        28   AE   9E 00022        MOVAB   DATE_BUF, DATE_DESC+4                   ; 0291
                                   7E   7C 00027        CLRQ    -(SP)                                   ; 0296
                              48   AE   9F 00029        PUSHAB  DATE_DESC
                                   7E   D4 0002C        CLRL    -(SP)
                66                 04   FB 0002E        CALLS   #4, SYS$ASCTIM
                2F  AE        31   AE   B0 00031        MOVW    DATE_BUF+9, DATE_BUF+7                  ; 0301
                20        28   AE   91 00036        CMPB    DATE_BUF, #32                           ; 0307
                                   04   12 0003A        BNEQ    1$
                28  AE             30   90 0003C        MOVB    #48, DATE_BUF
                2C  AE      2020   8F   A8 00040 1$:     BISW2   #8224, DATE_BUF+5                       ; 0314
                              28   AE   9F 00046        PUSHAB  DATE_BUF                                ; 0318
                                 016A   31 00049        BRW     8$
                30  AE 582D3030   8F   D0 0004C 2$:     MOVL    #1479356464, DATE_BUF                   ; 0345
                34  AE 302D5858   8F   D0 00054        MOVL    #808278104, DATE_BUF+4                  ; 0349
                38  AE             30   90 0005C        MOVB    #48, DATE_BUF+8                         ; 0353
          7E        00             54   01   7A 00060        EMUL    #1, R4, #0, -(SP)                       ; 0360
          53        53           8E 000003E8   8F   7B 00065        EDIV    #1000, (SP)+, DAY, DAY
                    52             54 000003E8   8F   C7 0006E        DIVL3   #1000, R4, R2                   ; 0361
                                   52      07B2   C2   9E 00076        MOVAB   1970(R2), YEAR
                                                 54   D5 0007B        TSTL    R4                              ; 0362
                                                 03   18 0007D        BGEQ    4$
                                              0131   31 0007F 3$:     BRW     7$
                           0000016E   8F             53   D1 00082 4$:     CMPL    DAY, #366
                                                 F4   14 00089        BGTR    3$
                                                 0E   12 0008B        BNEQ    5$                              ; 0368
          7E            00             52         01   7A 0008D        EMUL    #1, YEAR, #0, -(SP)             ; 0371
          50            50           8E           04   7B 00092        EDIV    #4, (SP)+, R0, R0
                                                 50   D5 00097        TSTL    R0
                                                 E4   12 00099        BNEQ    3$
                              30   AE 4A2D3130   8F   D0 0009B 5$:     MOVL    #1244475696, DATE_BUF           ; 0383
                              34   AE      4E41   8F   B0 000A3        MOVW    #20033, DATE_BUF+4              ; 0387
                              36   AE             2D   90 000A9        MOVB    #45, DATE_BUF+6                 ; 0389
                    50             52 000003E8   8F   C7 000AD        DIVL3   #1000, YEAR, R0                 ; 0390
                37  AE             50           30   81 000B5        ADDB3   #48, R0, DATE_BUF+7
                    50             52 00000064   8F   C7 000BA        DIVL3   #100, YEAR, R0                  ; 0391
          7E        00             50           01   7A 000C2        EMUL    #1, R0, #0, -(SP)
          50        50           8E             0A   7B 000C7        EDIV    #10, (SP)+, R0, R0
                38  AE             50           30   81 000CC        ADDB3   #48, R0, DATE_BUF+8
                    50             52           0A   C7 000D1        DIVL3   #10, YEAR, R0                   ; 0392
          7E        00             50           01   7A 000D5        EMUL    #1, R0, #0, -(SP)
          50        50           8E             0A   7B 000DA        EDIV    #10, (SP)+, R0, R0
                39  AE             50           30   81 000DF        ADDB3   #48, R0, DATE_BUF+9
```

```
                7E      00              52              01  7A 000E4           EMUL    #1, YEAR, #0, -(SP)                             ; 0393
                50      50              8E              0A  7B 000E9           EDIV    #10, (SP)+, R0, R0                              ;
                    3A  AE              50              30  81 000EE           ADDB3   #48, R0, DATE_BUF+10                            ;
                                    3B  AE  3A303020    8F  D0 000F3           MOVL    #976236576, DATE_BUF+11                         ; 0394
                                    3F  AE  303A3030    8F  D0 000FB           MOVL    #809119792, DATE_BUF+15                         ; 0398
                                    43  AE  30302E30    8F  D0 00103           MOVL    #808463920, DATE_BUF+19                         ; 0402
                                    47  AE              20  90 0010B           MOVB    #32, DATE_BUF+23                                ; 0406
                                    28  AE  010E0018    8F  D0 0010F           MOVL    #17694744, DATE_DESC                            ; 0410
                                    2C  AE          30  AE  9E 00117           MOVAB   DATE_BUF, DATE_DESC+4                           ; 0413
                                                    18  AE  9F 0011C           PUSHAB  Q_BASE_DATE                                     ; 0414
                                                    2C  AE  9F 0011F           PUSHAB  DATE_DESC                                       ;
                                        65              02  FB 00122           CALLS   #2, SYS$BINTIM                                  ;
                                    04  AE              30  90 00125           MOVB    #48, DAY_BUF                                    ; 0420
                                        50          FF  A4  9E 00129           MOVAB   -1(R4), R0                                      ; 0421
                    52                  50  00000064    8F  C7 0012D           DIVL3   #100, R0, R2                                    ;
                7E      00              52              01  7A 00135           EMUL    #1, R2, #0, -(SP)                               ;
                52      52              8E              0A  7B 0013A           EDIV    #10, (SP)+, R2, R2                              ;
                    05  AE              52              30  81 0013F           ADDB3   #48, R2, DAY_BUF+1                              ;
                        52              50              0A  C7 00144           DIVL3   #10, R0, R2                                     ;
                7E      00              52              01  7A 00148           EMUL    #1, R2, #0, -(SP)                               ; 0422
                52      52              8E              0A  7B 0014D           EDIV    #10, (SP)+, R2, R2                              ;
                    06  AE              52              30  81 00152           ADDB3   #48, R2, DAY_BUF+2                              ;
                7E      00              50              01  7A 00157           EMUL    #1, R0, #0, -(SP)                               ; 0423
                50      50              8E              0A  7B 0015C           EDIV    #10, (SP)+, R0, R0                              ;
                    07  AE              50              30  81 00161           ADDB3   #48, R0, DAY_BUF+3                              ;
                                    20  AE  010E0004    8F  D0 00166           MOVL    #17694724, DAY_DESC                             ; 0424
                                    24  AE          04  AE  9E 0016E           MOVAB   DAY_BUF, DAY_DESC+4                             ; 0427
                                                    10  AE  9F 00173           PUSHAB  Q_DELTA_DAYS                                    ; 0428
                                                    24  AE  9F 00176           PUSHAB  DAY_DESC                                        ;
                                        65              02  FB 00179           CALLS   #2, SYS$BINTIM                                  ;
                                        6E              02  D0 0017C           MOVL    #2, (SP)                                        ; 0434
                                                        5E  DD 0017F           PUSHL   SP                                              ;
                                                    0C  AE  9F 00181           PUSHAB  Q_FINAL_DATE                                    ;
                                                    18  AE  9F 00184           PUSHAB  Q_DELTA_DAYS                                    ;
                                                    24  AE  9F 00187           PUSHAB  Q_BASE_DATE                                     ;
                                00000000G           00      04  FB 0018A       CALLS   #4, LIB$SUBX                                    ;
                                                            7E  D4 00191       CLRL    -(SP)                                           ; 0438
                                                    0C  AE  9F 00193           PUSHAB  Q_FINAL_DATE                                    ;
                                                    30  AE  9F 00196           PUSHAB  DATE_DESC                                       ;
                                                            7E  D4 00199       CLRL    -(SP)                                           ;
                                        66                  04  FB 0019B       CALLS   #4, SYS$ASCTIM                                  ;
                                    37  AE          39  AE  B0 0019E           MOVW    DATE_BUF+9, DATE_BUF+7                          ; 0443
                                        20          30  AE  91 001A3           CMPB    DATE_BUF, #32                                   ; 0446
                                                            04  12 001A7       BNEQ    6$                                              ;
                                    30  AE                  30  90 001A9       MOVB    #48, DATE_BUF                                   ;
                                    34  AE          2020    8F  A8 001AD  6$:    BISW2   #8224, DATE_BUF+5                               ; 0449
                                                    30  AE  9F 001B3  7$:    PUSHAB  DATE_BUF                                        ; 0450
                                    04  AE                  09  D0 001B6  8$:    MOVL    #9, 4(SP)                                       ;
                                                    04  AE  9F 001BA           PUSHAB  4(SP)                                           ;
                                                    04  AC  DD 001BD           PUSHL   DATE_STR                                        ;
                                00000000G           00      03  FB 001C0       CALLS   #3, STR$COPY_R                                  ;
                                                                04 001C7       RET                                                     ; 0453

; Routine Size:  456 bytes,    Routine Base:  _BAS$CODE + 0000

;  362          0454  1
```

```
;  364     0455  1 GLOBAL ROUTINE BAS$TIME_T (                      ! Perform a TIME$ function
;  365     0456  1         TIME_STR,                                 ! Resulting string
;  366     0457  1         TIMENO                                    ! The time number, as defined below
;  367     0458  1     ) : NOVALUE =
;  368     0459  1
;  369     0460  1 !++
;  370     0461  1 ! FUNCTIONAL DESCRIPTION:
;  371     0462  1 !
;  372     0463  1 !       Perform a TIME$ function, as follows:
;  373     0464  1 !
;  374     0465  1 !       TIME$(0%) returns the current time of day in the form hh:mm
;  375     0466  1 !       TIME$(n%) returns the time corresponding to time number
;  376     0467  1 !               n, where n is the number of minutes before midnight.
;  377     0468  1 !
;  378     0469  1 ! FORMAL PARAMETERS:
;  379     0470  1 !
;  380     0471  1 !       TIME_STR.wt.d   The result string
;  381     0472  1 !       TIMENO.rl.v     The time number, or zero.
;  382     0473  1 !
;  383     0474  1 ! IMPLICIT INPUTS:
;  384     0475  1 !
;  385     0476  1 !       The system date and time, if TIMENO = 0.
;  386     0477  1 !
;  387     0478  1 ! IMPLICIT OUTPUTS:
;  388     0479  1 !
;  389     0480  1 !       NONE
;  390     0481  1 !
;  391     0482  1 ! ROUTINE VALUE:
;  392     0483  1 ! COMPLETION CODES:
;  393     0484  1 !
;  394     0485  1 !       NONE
;  395     0486  1 !
;  396     0487  1 ! SIDE EFFECTS:
;  397     0488  1 !
;  398     0489  1 !       NONE
;  399     0490  1 !
;  400     0491  1 !--
;  401     0492  1
;  402     0493  2     BEGIN
;  403     0494  2 !+
;  404     0495  2 ! If the time number is zero, get the system day and time, and reformat
;  405     0496  2 ! it to conform to BASIC-PLUS-2's standard.
;  406     0497  2 !-
;  407     0498  2
;  408     0499  3     IF (.TIMENO EQL 0)
;  409     0500  2     THEN
;  410     0501  3         BEGIN
;  411     0502  3
;  412     0503  3         LOCAL
;  413     0504  3             TIME_DESC : BLOCK [8, BYTE],
;  414     0505  3             TIME_BUF : VECTOR [24, BYTE],
;  415     0506  3             HOURS;
;  416     0507  3
;  417     0508  3 !+
;  418     0509  3 ! Build the descriptor for the string which will contain today's date and time.
;  419     0510  3 !-
;  420     0511  3         TIME_DESC [DSC$W_LENGTH] = 11;
```

```
;  421          0512  3          TIME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  422          0513  3          TIME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  423          0514  3          TIME_DESC [DSC$A_POINTER] = TIME_BUF;
;  424          0515  3
;  425          0516  3      !+
;  426          0517  3      ! get the current time
;  427          0518  3      !-
;  428          0519  3          $ASCTIM ( TIMBUF = TIME_DESC, CVTFLG = 1 );
;  429          0520  3
;  430          0521  3      !+
;  431          0522  3      ! Convert from 24-hour time to AM/PM.  We do this by appending " AM" to the
;  432          0523  3      ! time part of the string and then correcting it if the hours are between
;  433          0524  3      ! 12 and 23.  There is also a special test for 0 hours.
;  434          0525  3      !-
;  435          0526  3          TIME_BUF [5] = %C' ';
;  436          0527  3          TIME_BUF [6] = %C'A';
;  437          0528  3          TIME_BUF [7] = %C'M';
;  438          0529  3          HOURS = ((.TIME_BUF [0] - %C'0')*10) + (.TIME_BUF [1] - %C'0');
;  439          0530  3
;  440          0531  3          SELECTONE .HOURS OF
;  441          0532  3              SET
;  442          0533  3
;  443          0534  3              [0] :
;  444          0535  4                  BEGIN                          ! It is AM, but we must change zero hours to 12.
;  445          0536  4                  TIME_BUF [0] = %C'1';
;  446          0537  4                  TIME_BUF [1] = %C'2';
;  447          0538  3                  END;
;  448          0539  3
;  449          0540  3              [1 TO 11] :
;  450          0541  4                  BEGIN                          ! It is AM, nothing special to do here.
;  451          0542  4                  0
;  452          0543  3                  END;
;  453          0544  3
;  454          0545  3              [12] :
;  455          0546  4                  BEGIN                          ! It is PM, but the hour must not be corrected.
;  456          0547  4                  TIME_BUF [6] = %C'P';
;  457          0548  3                  END;
;  458          0549  3
;  459          0550  3              [13 TO 23] :
;  460          0551  4                  BEGIN                          ! It is PM, and the hour must be corrected.
;  461          0552  4                  HOURS = .HOURS - 12;
;  462          0553  4                  TIME_BUF [0] = (.HOURS/10) + %C'0';
;  463          0554  4                  TIME_BUF [1] = (.HOURS MOD 10) + %C'0';
;  464          0555  4                  TIME_BUF [6] = %C'P'
;  465          0556  3                  END;
;  466          0557  3
;  467          0558  3              [OTHERWISE] :
;  468          0559  4                  BEGIN                          ! The time is quite unreasonable.  Give a fatal error.
;  469          0560  4                  LIB$STOP (OTS$_FATINTERR);
;  470          0561  3                  END;
;  471          0562  3              TES;
;  472          0563  3
;  473          0564  3      !+
;  474          0565  3      ! Return the string to the user.
;  475          0566  3      !-
;  476          0567  3          STR$COPY_R (.TIME_STR, %REF (8), TIME_BUF [0]);
;  477          0568  3          RETURN;
```

```
;  478          0569  2          END;
;  479          0570  2
;  480          0571  2      !+
;  481          0572  2      ! Come here if the time argument is not zero.  We must compute the
;  482          0573  2      ! time corresponding to TIMENO minutes before midnight.
;  483          0574  2      !-
;  484          0575  3          BEGIN
;  485          0576  3
;  486          0577  3          LOCAL
;  487          0578  3              HOURS,
;  488          0579  3              MINUTES,
;  489          0580  3              TIME_BUF : VECTOR [8, BYTE],
;  490          0581  3              AMPM;
;  491          0582  3      !+
;  492          0583  3      ! Allow arguments in the range of 0 to 1440 (24 hrs
;  493          0584  3      ! before midnight) only.
;  494          0585  3      !-
;  495          0586  3
;  496          0587  3          SELECTONE (.TIMENO) OF
;  497          0588  3              SET
;  498          0589  3
;  499          0590  3              [0 TO 1440] :
;  500          0591  3              ;
;  501          0592  3
;  502          0593  3              [OTHERWISE] :
;  503          0594  3                  BAS$$STOP (BAS$K_ARGOUTBOU) ;
;  504          0595  3              TES;
;  505          0596  3
;  506          0597  3          MINUTES = (24*60) - .TIMENO;
;  507          0598  3          HOURS = .MINUTES/60;
;  508          0599  3          MINUTES = .MINUTES - (.HOURS*60);
;  509          0600  3          AMPM = %C'A';
;  510          0601  3
;  511          0602  3          SELECTONE (.HOURS) OF
;  512          0603  3              SET
;  513          0604  3
;  514          0605  3              [0] :
;  515          0606  3                  HOURS = 12;
;  516          0607  3
;  517          0608  3              [1 TO 11] :
;  518          0609  3              ;
;  519          0610  3
;  520          0611  3              [12] :
;  521          0612  3                  AMPM = %C'P';
;  522          0613  3
;  523          0614  3              [13 TO 23] :
;  524          0615  4                  BEGIN
;  525          0616  4                  HOURS = .HOURS - 12;
;  526          0617  4                  AMPM = %C'P';
;  527          0618  3                  END;
;  528          0619  3
;  529          0620  3              [OTHERWISE] :
;  530          0621  3                  LIB$STOP (OTS$_FATINTERR);
;  531          0622  3              TES;
;  532          0623  3
;  533          0624  3      !+
;  534          0625  3      ! Now store the time in the time buffer.
```

```
;  535          0626  3 !-
;  536          0627  3     TIME_BUF [0] = ((.HOURS/10) + %C'0');
;  537          0628  3     TIME_BUF [1] = ((.HOURS MOD 10) + %C'0');
;  538          0629  3     TIME_BUF [2] = %C':';
;  539          0630  3     TIME_BUF [3] = ((.MINUTES/10) + %C'0');
;  540          0631  3     TIME_BUF [4] = ((.MINUTES MOD 10) + %C'0');
;  541          0632  3     TIME_BUF [5] = %C' ';
;  542          0633  3     TIME_BUF [6] = .AMPM;
;  543          0634  3     TIME_BUF [7] = %C'M';
;  544          0635  3 !+
;  545          0636  3 ! Convey the buffer to the user's string.
;  546          0637  3 !-
;  547          0638  3     STR$COPY_R (.TIME_STR, %REF (8), TIME_BUF [0]);
;  548          0639  3     RETURN;
;  549          0640  2     END;
;  550          0641  1     END;                                    ! end of BAS$TIME_T


                                 007C 00000         .ENTRY  BAS$TIME_T, Save R2,R3,R4,R5,R6                 ; 0455
                      56 00000000G   9E 00002       MOVAB   LIB$STOP, R6                                    ;
                      55 00000000G   D0 00009       MOVL    #OTS$_FATINTERR, R5                             ;
                      5E             C2 00010       SUBL2   #36, SP                                         ;
                      53       08    D0 00013       MOVL    TIMENO, R3                                      ; 0499
                                     13 00017       BEQL    1$                                              ;
                                0088 31 00019       BRW     7$                                              ;
                   1C AE 010E000B    D0 0001C 1$:    MOVL    #17694731, TIME_DESC                            ; 0511
                   20 AE       04    9E 00024       MOVAB   TIME_BUF, TIME_DESC+4                           ; 0514
                                     DD 00029       PUSHL   #1                                              ; 0519
                                     D4 0002B       CLRL    -(SP)                                           ;
                               24 AE 9F 0002D       PUSHAB  TIME_DESC                                       ;
                                     D4 00030       CLRL    -(SP)                                           ;
                00000000G      00    FB 00032       CALLS   #4, SYS$ASCTIM                                  ;
                        09 AE 4120    B0 00039       MOVW    #16672, TIME_BUF+5                              ; 0526
                        0B AE   4D    90 0003F       MOVB    #77, TIME_BUF+7                                 ; 0528
                      50       04 AE 9A 00044       MOVZBL  TIME_BUF, R0                                    ; 0529
                      50          0A C4 00048       MULL2   #10, R0                                         ;
                      51       05 AE 9A 0004B       MOVZBL  TIME_BUF+1, R1                                  ;
                      50          51 C0 0004F       ADDL2   R1, R0                                          ;
                      50    FDF0  C0 9E 00052       MOVAB   -528(R0), HOURS                                 ;
                                     12 00057       BNEQ    2$                                              ; 0534
                   04 AE    3231    B0 00059       MOVW    #12849, TIME_BUF                                ; 0536
                                     11 0005F       BRB     6$                                              ; 0531
                                     15 00061 2$:   BLEQ    3$                                              ; 0540
                      0B             D1 00063       CMPL    HOURS, #11                                      ;
                                     15 00066       BLEQ    6$                                              ;
                      0C             D1 00068 3$:   CMPL    HOURS, #12                                      ; 0545
                                     13 0006B       BEQL    4$                                              ;
                      0D             D1 0006D       CMPL    HOURS, #13                                      ; 0550
                                     19 00070       BLSS    5$                                              ;
                      17             D1 00072       CMPL    HOURS, #23                                      ;
                                     14 00075       BGTR    5$                                              ;
                      50             C2 00077       SUBL2   #12, HOURS                                      ; 0552
                   51    50          C7 0007A       DIVL3   #10, HOURS, R1                                  ; 0553
             04 AE       51          81 0007E       ADDB3   #48, R1, TIME_BUF                               ;
```

```
 7E        00                50              01  7A 00083        EMUL     #1, HOURS, #0, -(SP)               ; 0554
 50        50                8E              0A  7B 00088        EDIV     #10, (SP)+, R0, R0                 ;
        05 AE                50              30  81 0008D        ADDB3    #48, R0, TIME_BUF+1                ;
                       0A    AE       50     8F  90 00092 4$:    MOVB     #80, TIME_BUF+6                    ; 0555
                                             05  11 00097        BRB      6$                                 ;
                                             55  DD 00099 5$:    PUSHL    R5                                 ; 0560
                             66              01  FB 0009B        CALLS    #1, LIB$STOP                       ;
                                      04     AE  9F 0009E 6$:    PUSHAB   TIME_BUF                           ; 0567
                                          009E   31 000A1        BRW      15$                                ;
                                             09  19 000A4 7$:    BLSS     8$                                 ; 0590
                000005A0     8F              53  D1 000A6        CMPL     R3, #1440                          ;
                                             0B  15 000AD        BLEQ     9$                                 ;
                             7E       00G    8F  9A 000AF 8$:    MOVZBL   #BAS$K_ARGOUTBOU, -(SP)            ; 0594
                00000000G    00              01  FB 000B3        CALLS    #1, BAS$$STOP                      ;
           53   000005A0     8F              53  C3 000BA 9$:    SUBL3    R3, #1440, MINUTES                 ; 0597
           52                53              3C  C7 000C2        DIVL3    #60, MINUTES, HOURS                ; 0598
           50                52              3C  C5 000C6        MULL3    #60, HOURS, R0                     ; 0599
                             53              50  C2 000CA        SUBL2    R0, MINUTES                        ;
                             54       41     8F  9A 000CD        MOVZBL   #65, AMPM                          ; 0600
                                             52  D5 000D1        TSTL     HOURS                              ; 0605
                                             05  12 000D3        BNEQ     10$                                ;
                             52              0C  D0 000D5        MOVL     #12, HOURS                         ; 0606
                                             24  11 000D8        BRB      14$                                ;
                                             05  15 000DA 10$:   BLEQ     11$                                ; 0608
                             0B              52  D1 000DC        CMPL     HOURS, #11                         ;
                                             1D  15 000DF        BLEQ     14$                                ;
                             0C              52  D1 000E1 11$:   CMPL     HOURS, #12                         ; 0611
                                             0D  13 000E4        BEQL     12$                                ;
                             0D              52  D1 000E6        CMPL     HOURS, #13                         ; 0614
                                             0E  19 000E9        BLSS     13$                                ;
                             17              52  D1 000EB        CMPL     HOURS, #23                         ;
                                             09  14 000EE        BGTR     13$                                ;
                             52              0C  C2 000F0        SUBL2    #12, HOURS                         ; 0616
                             54       50     8F  9A 000F3 12$:   MOVZBL   #80, AMPM                          ; 0617
                                             05  11 000F7        BRB      14$                                ; 0602
                                             55  DD 000F9 13$:   PUSHL    R5                                 ; 0621
                             66              01  FB 000FB        CALLS    #1, LIB$STOP                       ;
           50                52              0A  C7 000FE 14$:   DIVL3    #10, HOURS, R0                     ; 0627
        1C AE                50              30  81 00102        ADDB3    #48, R0, TIME_BUF                  ;
 7E        00                52              01  7A 00107        EMUL     #1, HOURS, #0, -(SP)               ; 0628
 50        50                8E              0A  7B 0010C        EDIV     #10, (SP)+, R0, R0                 ;
        1D AE                50              30  81 00111        ADDB3    #48, R0, TIME_BUF+1                ;
                       1E    AE              3A  90 00116        MOVB     #58, TIME_BUF+2                    ; 0629
           50                53              0A  C7 0011A        DIVL3    #10, MINUTES, R0                   ; 0630
        1F AE                50              30  81 0011E        ADDB3    #48, R0, TIME_BUF+3                ;
 7E        00                53              01  7A 00123        EMUL     #1, MINUTES, #0, -(SP)             ; 0631
 50        50                8E              0A  7B 00128        EDIV     #10, (SP)+, R0, R0                 ;
        20 AE                50              30  81 0012D        ADDB3    #48, R0, TIME_BUF+4                ;
                       21    AE              20  90 00132        MOVB     #32, TIME_BUF+5                    ; 0632
                       22    AE              54  90 00136        MOVB     AMPM, TIME_BUF+6                   ; 0633
                       23    AE       4D     8F  90 0013A        MOVB     #77, TIME_BUF+7                    ; 0634
                                      1C     AE  9F 0013F        PUSHAB   TIME_BUF                           ; 0638
                       04    AE              08  D0 00142 15$:   MOVL     #8, 4(SP)                          ;
                                      04     AE  9F 00146        PUSHAB   4(SP)                              ;
                                      04     AC  DD 00149        PUSHL    TIME_STR                           ;
                00000000G    00              03  FB 0014C        CALLS    #3, STR$COPY_R                     ;
                                                 04 00153        RET                                         ; 0641
```

```
; Routine Size:  340 bytes,    Routine Base:  _BAS$CODE + 01C8

;  551          0642  1
```

```
;  553     0643  1 GLOBAL ROUTINE BAS$TIME_F (                          ! Perform a TIME function
;  554     0644  1         TYPE                                         ! The type of time requested
;  555     0645  1     ) =
;  556     0646  1
;  557     0647  1 !++
;  558     0648  1 ! FUNCTIONAL DESCRIPTION:
;  559     0649  1 !
;  560     0650  1 !       Perform a TIME function, as follows:
;  561     0651  1 !
;  562     0652  1 !       TIME(0%) returns the current time of day
;  563     0653  1 !                in seconds since midnight
;  564     0654  1 !       TIME(1%) returns the CPU time used by the current job in
;  565     0655  1 !                tenths of a second (100-millisecond units)
;  566     0656  1 !       TIME(2%) returns the connect time for the current job in minutes
;  567     0657  1 !       TIME(3%) and TIME(4%) are not implemented on VAX/VMS.
;  568     0658  1 !
;  569     0659  1 !       Any other value of the argument is undefined.
;  570     0660  1 !
;  571     0661  1 ! FORMAL PARAMETERS:
;  572     0662  1 !
;  573     0663  1 !       TYPE.rl.v       The type of time requested, see above.
;  574     0664  1 !
;  575     0665  1 ! IMPLICIT INPUTS:
;  576     0666  1 !
;  577     0667  1 !       The system date and time, and other system timing statistics
;  578     0668  1 !
;  579     0669  1 ! IMPLICIT OUTPUTS:
;  580     0670  1 !
;  581     0671  1 !       NONE
;  582     0672  1 !
;  583     0673  1 ! ROUTINE VALUE:
;  584     0674  1 ! COMPLETION CODES:
;  585     0675  1 !
;  586     0676  1 !       The requested type of time, as a floating point number.
;  587     0677  1 !
;  588     0678  1 ! SIDE EFFECTS:
;  589     0679  1 !
;  590     0680  1 !       NONE
;  591     0681  1 !
;  592     0682  1 !--
;  593     0683  1
;  594     0684  2     BEGIN
;  595     0685  2
;  596     0686  2     BUILTIN
;  597     0687  2         CVTLF;
;  598     0688  2
;  599     0689  2 !+
;  600     0690  2 ! Compute the requested time value, based on the input argument.
;  601     0691  2 !-
;  602     0692  2
;  603     0693  2     CASE (.TYPE) FROM 0 TO 4 OF
;  604     0694  2         SET
;  605     0695  2
;  606     0696  2         [0] :
;  607     0697  3             BEGIN
;  608     0698  3
;  609     0699  3             LOCAL
```

```
;  610     0700  3                  TIME_DESC : BLOCK [8, BYTE],
;  611     0701  3                  TIME_BUF : VECTOR [24, BYTE],
;  612     0702  3                  HOURS,
;  613     0703  3                  MINUTES,
;  614     0704  3                  SECONDS,
;  615     0705  3                  RESULT;
;  616     0706  3
;  617     0707  3          !+
;  618     0708  3          ! Set up the buffer descriptor
;  619     0709  3          !-
;  620     0710  3                  TIME_DESC [DSC$W_LENGTH] = 11;
;  621     0711  3                  TIME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  622     0712  3                  TIME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  623     0713  3                  TIME_DESC [DSC$A_POINTER] = TIME_BUF;
;  624     0714  3          !+
;  625     0715  3          ! Extract the current time from the system.
;  626     0716  3          !-
;  627     0717  3                  $ASCTIM ( TIMBUF = TIME_DESC, CVTFLG = 1 );
;  628     0718  3
;  629     0719  3          !+
;  630     0720  3          ! Extract from the character string the number of seconds since midnight.
;  631     0721  3          !-
;  632     0722  3                  HOURS = ((.TIME_BUF [0] - %C'0')*10) + (.TIME_BUF [1] - %C'0');
;  633     0723  3                  MINUTES = ((.TIME_BUF [3] - %C'0')*10) + (.TIME_BUF [4] - %C'0');
;  634     0724  3                  SECONDS = ((.TIME_BUF [6] - %C'0')*10) + (.TIME_BUF [7] - %C'0');
;  635     0725  3          !+
;  636     0726  3          ! Return the number of seconds since midnight, as a floating point number.
;  637     0727  3          !-
;  638     0728  3                  CVTLF (%REF (.SECONDS + (.MINUTES*60) + (.HOURS*60*60)), RESULT);
;  639     0729  3                  RETURN (.RESULT);
;  640     0730  3                  END;
;  641     0731  2
;  642     0732  2              [1] :
;  643     0733  3                  BEGIN
;  644     0734  3
;  645     0735  3                  LOCAL
;  646     0736  3                      GETJPI_LIST : BLOCK [16, BYTE],
;  647     0737  3                      L_CPU_TIME  : VOLATILE,
;  648     0738  3                      RESULT;
;  649     0739  3
;  650     0740  3          !+
;  651     0741  3          ! Fill in the GETJPI list.
;  652     0742  3          !-
;  653     0743  3                  GETJPI_LIST [0, 0, 16, 0] = 4;      ! Buffer length
;  654     0744  3                  GETJPI_LIST [2, 0, 16, 0] = JPI$_CPUTIM;    ! Get CPU time
;  655     0745  3                  GETJPI_LIST [4, 0, %BPVAL, 0] = L_CPU_TIME;
;  656     0746  3                  GETJPI_LIST [8, 0, %BPVAL, 0] = 0;  ! Don't return length
;  657     0747  3                  GETJPI_LIST [12, 0, %BPVAL, 0] = 0; ! That's all
;  658     0748  3          !+
;  659     0749  3          ! Get the information from the system.
;  660     0750  3          !-
;  661     0751  4                  BEGIN
;  662     0752  4                  LOCAL
;  663     0753  4                      EVENT_FLAG,
;  664     0754  4                      STATUS;
;  665     0755  4
;  666     0756  4                  STATUS = LIB$GET_EF (EVENT_FLAG);
```

```
;  667     0757  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  668     0758  4
;  669     0759  4              STATUS = $GETJPI (EFN = .EVENT_FLAG, ITMLST = GETJPI_LIST);
;  670     0760  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  671     0761  4
;  672     0762  4              STATUS = LIB$FREE_EF (EVENT_FLAG);
;  673     0763  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  674     0764  4
;  675     0765  3              END;
;  676     0766  3          !+
;  677     0767  3          ! Convert the 10-millisecond units into 100-millisecond units.
;  678     0768  3          !-
;  679     0769  3              CVTLF (%REF (.L_CPU_TIME/10), RESULT);
;  680     0770  3          !+
;  681     0771  3          ! Return the CPU time in 100-millisecond units as a floating point number
;  682     0772  3          !-
;  683     0773  3              RETURN (.RESULT);
;  684     0774  2              END;
;  685     0775  2
;  686     0776  2          [2] :
;  687     0777  3              BEGIN
;  688     0778  3
;  689     0779  3              LOCAL
;  690     0780  3                  GETJPI_LIST : BLOCK [16, BYTE],
;  691     0781  3                  Q_CONN_TIME : VECTOR [2],
;  692     0782  3                  Q_LOGIN_TIME : VECTOR [2],
;  693     0783  3                  Q_NOW : VECTOR [2],
;  694     0784  3                  RESULT,
;  695     0785  3                  TIME_DESC : BLOCK [8, BYTE],
;  696     0786  3                  TIME_BUF : VECTOR [17, BYTE],
;  697     0787  3                  DAYS,
;  698     0788  3                  HOURS,
;  699     0789  3                  MINUTES,
;  700     0790  3                  STATUS,
;  701     0791  3                  EVENT_FLAG;
;  702     0792  3
;  703     0793  3          !+
;  704     0794  3          ! Fill in the GETJPI list.
;  705     0795  3          !-
;  706     0796  3                  GETJPI_LIST [0, 0, 16, 0] = 8;       ! Buffer length
;  707     0797  3                  GETJPI_LIST [2, 0, 16, 0] = JPI$_LOGINTIM;  ! Login time
;  708     0798  3                  GETJPI_LIST [4, 0, %BPVAL, 0] = Q_LOGIN_TIME;
;  709     0799  3                  GETJPI_LIST [8, 0, %BPVAL, 0] = 0;  ! Don't return length
;  710     0800  3                  GETJPI_LIST [12, 0, %BPVAL, 0] = 0; ! That's all
;  711     0801  3
;  712     0802  3          !+
;  713     0803  3          ! Allocate an event flag.
;  714     0804  3          !-
;  715     0805  3                  STATUS = LIB$GET_EF (EVENT_FLAG);
;  716     0806  3                  IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  717     0807  3          !+
;  718     0808  3          ! Get the information from the system.
;  719     0809  3          !-
;  720     0810  3                  STATUS = $GETJPI (EFN = .EVENT_FLAG, ITMLST = GETJPI_LIST);
;  721     0811  3                  IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  722     0812  3
;  723     0813  3          !+
```

```
;  724          0814  3 ! Deallocate the event flag.
;  725          0815  3 !-
;  726          0816  3             STATUS = LIB$FREE_EF (EVENT_FLAG);
;  727          0817  3             IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  728          0818  3
;  729          0819  3 !+
;  730          0820  3 ! Now get the current time from the system.
;  731          0821  3 !-
;  732          0822  3             $GETTIM (TIMADR = Q_NOW);
;  733          0823  3 !+
;  734          0824  3 ! Subtract the two to get the execution time.  This must be done
;  735          0825  3 ! using quadword arithmetic.
;  736          0826  3 !-
;  737          0827  3             LIB$SUBX (Q_LOGIN_TIME, Q_NOW, Q_CONN_TIME, %REF (2));
;  738          0828  3 !+
;  739          0829  3 ! Use the $ASCTIM system service to convert the 64-bit connect time
;  740          0830  3 ! to a number of days, hours, minutes and seconds.
;  741          0831  3 !-
;  742          0832  3             TIME_DESC [DSC$W_LENGTH] = 17;
;  743          0833  3             TIME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  744          0834  3             TIME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  745          0835  3             TIME_DESC [DSC$A_POINTER] = TIME_BUF;
;  746          0836  3             $ASCTIM (TIMBUF = TIME_DESC, TIMADR = Q_CONN_TIME);
;  747          0837  3 !+
;  748          0838  3 ! Turn leading blanks in the number of days into zeros.
;  749          0839  3 !-
;  750          0840  3
;  751          0841  3             INCR COUNTER FROM 0 TO 3 DO
;  752          0842  3
;  753          0843  3                 IF (.TIME_BUF [.COUNTER] EQL %C' ') THEN TIME_BUF [.COUNTER] = %C'0';
;  754          0844  3
;  755          0845  3 !+
;  756          0846  3 ! Convert the string into a number of minutes.
;  757          0847  3 !-
;  758          0848  3             DAYS = ((.TIME_BUF [0] - %C'0')*1000) +     !
;  759          0849  3             ((.TIME_BUF [1] - %C'0')*100) +     !
;  760          0850  3             ((.TIME_BUF [2] - %C'0')*10) +      !
;  761          0851  3             (.TIME_BUF [3] - %C'0');
;  762          0852  3             HOURS = ((.TIME_BUF [5] - %C'0')*10) + (.TIME_BUF [6] - %C'0');
;  763          0853  3             MINUTES = ((.TIME_BUF [8] - %C'0')*10) + (.TIME_BUF [9] - %C'0');
;  764          0854  3 !+
;  765          0855  3 ! Convert the total number of minutes to floating point.
;  766          0856  3 !-
;  767          0857  3             CVTLF (%REF (.MINUTES + (.HOURS*60) + (.DAYS*24*60)), RESULT);
;  768          0858  3 !+
;  769          0859  3 ! Return the connect time in minutes as a floating point number.
;  770          0860  3 !-
;  771          0861  3             RETURN (.RESULT);
;  772          0862  2             END;
;  773          0863  2
;  774          0864  2         [3, 4] :
;  775          0865  2 !+
;  776          0866  2 ! These functions are not implemented.  They returned the kilo-core
;  777          0867  2 ! ticks and device time in RSTS.  For compatability, return zero.
;  778          0868  2 !-
;  779          0869  2             RETURN (0);
;  780          0870  2
```

```
;  781          0871  2          [OUTRANGE] :
;  782          0872  2      !+
;  783          0873  2      ! Other values have never been implemented in anything we must be
;  784          0874  2      ! compatable with, so give an error message.
;  785          0875  2      !-
;  786          0876  2          BAS$$STOP (BAS$K_NOTIMP);
;  787          0877  2          TES;
;  788          0878  2
;  789          0879  2      !+
;  790          0880  2      ! In the unlikely event that the above CASE expression falls through,
;  791          0881  2      ! return a zero.
;  792          0882  2      !-
;  793          0883  2          RETURN (0);
;  794          0884  1          END;                                    ! end of BAS$TIME_F


                                                    .EXTRN  SYS$GETJPI, SYS$GETTIM

                                          01FC 00000    .ENTRY  BAS$TIME_F, Save R2,R3,R4,R5,R6,R7,R8       ; 0643
                                58 00000000G   00 9E 00002    MOVAB   LIB$FREE_EF, R8                        ;
                                57 00000000G   00 9E 00009    MOVAB   SYS$GETJPI, R7                         ;
                                56 00000000G   00 9E 00010    MOVAB   LIB$GET_EF, R6                         ;
                                55 00000000G   00 9E 00017    MOVAB   SYS$ASCTIM, R5                         ;
                                54 00000000G   00 9E 0001E    MOVAB   LIB$STOP, R4                           ;
                                5E       B0    AE 9E 00025    MOVAB   -80(SP), SP                            ;
                          04    00       04    AC CF 00029    CASEL   TYPE, #0, #4                           ; 0693
                01D6      00CF      007B          0018 0002E 1$:  .WORD   2$-1$,-                            ;
                                                  01D6 00036              3$-1$,-                            ;
                                                                          7$-1$,-                            ;
                                                                          15$-1$,-                           ;
                                                                          15$-1$                             ;
                          7E       00G   8F 9A 00038    MOVZBL  #BAS$K_NOTIMP, -(SP)                       ; 0876
00000000G                 00             01 FB 0003C    CALLS   #1, BAS$$STOP
                                      01BE   31 00043    BRW     15$
                    48    AE 010E000B    8F D0 00046 2$: MOVL    #17694731, TIME_DESC                       ; 0710
                    4C    AE       30    AE 9E 0004E    MOVAB   TIME_BUF, TIME_DESC+4                      ; 0713
                                         01 DD 00053    PUSHL   #1                                         ; 0717
                                         7E D4 00055    CLRL    -(SP)
                                      50 AE 9F 00057    PUSHAB  TIME_DESC
                                         7E D4 0005A    CLRL    -(SP)
                                65       04 FB 0005C    CALLS   #4, SYS$ASCTIM
                                52       30 AE 9A 0005F    MOVZBL  TIME_BUF, R2                           ; 0722
                                52       0A C4 00063    MULL2   #10, R2
                                50       31 AE 9A 00066    MOVZBL  TIME_BUF+1, R0
                                52       50 C0 0006A    ADDL2   R0, R2
                                52     FDF0 C2 9E 0006D    MOVAB   -528(R2), HOURS
                                51       33 AE 9A 00072    MOVZBL  TIME_BUF+3, R1                         ; 0723
                                51       0A C4 00076    MULL2   #10, R1
                                50       34 AE 9A 00079    MOVZBL  TIME_BUF+4, R0
                                51       50 C0 0007D    ADDL2   R0, R1
                                51     FDF0 C1 9E 00080    MOVAB   -528(R1), MINUTES
                                50       36 AE 9A 00085    MOVZBL  TIME_BUF+6, R0                         ; 0724
                                50       0A C4 00089    MULL2   #10, R0
                                53       37 AE 9A 0008C    MOVZBL  TIME_BUF+7, R3
                                50       53 C0 00090    ADDL2   R3, R0
                                50     FDF1 C0 9E 00093    MOVAB   -527(R0), SECONDS
```

```
                        51              3C  C4 00098          MULL2   #60, R1                                        ; 0728
                        50          7041  9E 0009B          MOVAB   -(SECONDS)[R1], R0                              ;
                        52  00000E10   8F  C4 0009F          MULL2   #3600, R2                                       ;
                                    0151  31 000A6          BRW     13$                                             ;
                40      AE  04070004   8F  D0 000A9  3$:     MOVL    #67567620, GETJPI_LIST                          ; 0743
                44      AE          3C  AE 9E 000B1          MOVAB   L_CPU_TIME, GETJPI_LIST+4                       ; 0745
                                    48  AE 7C 000B6          CLRQ    GETJPI_LIST+8                                   ; 0746
                                    04  AE 9F 000B9          PUSHAB  EVENT_FLAG                                      ; 0756
                        66              01  FB 000BC          CALLS   #1, LIB$GET_EF                                  ;
                        52              50  D0 000BF          MOVL    R0, STATUS                                      ;
                        05              52  E8 000C2          BLBS    STATUS, 4$                                      ; 0757
                                        52  DD 000C5          PUSHL   STATUS                                          ;
                        64              01  FB 000C7          CALLS   #1, LIB$STOP                                    ;
                                        7E  7C 000CA  4$:     CLRQ    -(SP)                                           ; 0759
                                        7E  D4 000CC          CLRL    -(SP)                                           ;
                                    4C  AE  9F 000CE          PUSHAB  GETJPI_LIST                                     ;
                                        7E  7C 000D1          CLRQ    -(SP)                                           ;
                                    1C  AE  DD 000D3          PUSHL   EVENT_FLAG                                      ;
                        67              07  FB 000D6          CALLS   #7, SYS$GETJPI                                  ;
                        52              50  D0 000D9          MOVL    R0, STATUS                                      ;
                        05              52  E8 000DC          BLBS    STATUS, 5$                                      ; 0760
                                        52  DD 000DF          PUSHL   STATUS                                          ;
                        64              01  FB 000E1          CALLS   #1, LIB$STOP                                    ;
                                    04  AE  9F 000E4  5$:     PUSHAB  EVENT_FLAG                                      ; 0762
                        68              01  FB 000E7          CALLS   #1, LIB$FREE_EF                                 ;
                        52              50  D0 000EA          MOVL    R0, STATUS                                      ;
                        05              52  E8 000ED          BLBS    STATUS, 6$                                      ; 0763
                                        52  DD 000F0          PUSHL   STATUS                                          ;
                        64              01  FB 000F2          CALLS   #1, LIB$STOP                                    ;
        50      3C      AE              0A  C7 000F5  6$:     DIVL3   #10, L_CPU_TIME, R0                             ; 0769
                                    0100  31 000FA          BRW     14$                                             ;
                40      AE  02060008   8F  D0 000FD  7$:     MOVL    #33947656, GETJPI_LIST                          ; 0796
                44      AE          30  AE 9E 00105          MOVAB   Q_LOGIN_TIME, GETJPI_LIST+4                     ; 0798
                                    48  AE 7C 0010A          CLRQ    GETJPI_LIST+8                                   ; 0799
                                    08  AE 9F 0010D          PUSHAB  EVENT_FLAG                                      ; 0805
                        66              01  FB 00110          CALLS   #1, LIB$GET_EF                                  ;
                        52              50  D0 00113          MOVL    R0, STATUS                                      ;
                        05              52  E8 00116          BLBS    STATUS, 8$                                      ; 0806
                                        52  DD 00119          PUSHL   STATUS                                          ;
                        64              01  FB 0011B          CALLS   #1, LIB$STOP                                    ;
                                        7E  7C 0011E  8$:     CLRQ    -(SP)                                           ; 0810
                                        7E  D4 00120          CLRL    -(SP)                                           ;
                                    4C  AE  9F 00122          PUSHAB  GETJPI_LIST                                     ;
                                        7E  7C 00125          CLRQ    -(SP)                                           ;
                                    20  AE  DD 00127          PUSHL   EVENT_FLAG                                      ;
                        67              07  FB 0012A          CALLS   #7, SYS$GETJPI                                  ;
                        52              50  D0 0012D          MOVL    R0, STATUS                                      ;
                        05              52  E8 00130          BLBS    STATUS, 9$                                      ; 0811
                                        52  DD 00133          PUSHL   STATUS                                          ;
                        64              01  FB 00135          CALLS   #1, LIB$STOP                                    ;
                                    08  AE  9F 00138  9$:     PUSHAB  EVENT_FLAG                                      ; 0816
                        68              01  FB 0013B          CALLS   #1, LIB$FREE_EF                                 ;
                        52              50  D0 0013E          MOVL    R0, STATUS                                      ;
                        05              52  E8 00141          BLBS    STATUS, 10$                                     ; 0817
                                        52  DD 00144          PUSHL   STATUS                                          ;
                        64              01  FB 00146          CALLS   #1, LIB$STOP                                    ;
                                    28  AE  9F 00149  10$:    PUSHAB  Q_NOW                                           ; 0822
```

```
                00000000G  00              01  FB 0014C          CALLS   #1, SYS$GETTIM                           ;
                           6E               02  D0 00153          MOVL    #2, (SP)                                ; 0827
                                            5E  DD 00156          PUSHL   SP                                      ;
                                      3C    AE  9F 00158          PUSHAB  Q_CONN_TIME                             ;
                                      30    AE  9F 0015B          PUSHAB  Q_NOW                                   ;
                                      3C    AE  9F 0015E          PUSHAB  Q_LOGIN_TIME                            ;
                00000000G  00              04  FB 00161          CALLS   #4, LIB$SUBX                            ;
                      20   AE  010E0011     8F  D0 00168          MOVL    #17694737, TIME_DESC                    ; 0832
                      24   AE        0C     AE  9E 00170          MOVAB   TIME_BUF, TIME_DESC+4                   ; 0835
                                            7E  D4 00175          CLRL    -(SP)                                   ; 0836
                                      3C    AE  9F 00177          PUSHAB  Q_CONN_TIME                             ;
                                      28    AE  9F 0017A          PUSHAB  TIME_DESC                               ;
                                            7E  D4 0017D          CLRL    -(SP)                                   ;
                           65               04  FB 0017F          CALLS   #4, SYS$ASCTIM                          ;
                                            50  D4 00182          CLRL    COUNTER                                 ; 0841
                           20         0C AE40  91  00184 11$:      CMPB    TIME_BUF[COUNTER], #32                  ; 0843
                                            05  12  00189          BNEQ    12$                                     ;
                      0C AE40               30  90  0018B          MOVB    #48, TIME_BUF[COUNTER]                  ;
                F0         50               03  F3  00190 12$:     AOBLEQ  #3, COUNTER, 11$                        ;
                           50         0C    AE  9A  00194          MOVZBL  TIME_BUF, R0                            ; 0848
                           50  000003E8     8F  C4  00198          MULL2   #1000, R0                               ;
                           51         0D    AE  9A  0019F          MOVZBL  TIME_BUF+1, R1                          ; 0849
                           51  00000064     8F  C4  001A3          MULL2   #100, R1                                ;
                           50               51  C0  001AA          ADDL2   R1, R0                                  ; 0848
                           51         0E    AE  9A  001AD          MOVZBL  TIME_BUF+2, R1                          ; 0850
                           51               0A  C4  001B1          MULL2   #10, R1                                 ;
                52         50               51  C1  001B4          ADDL3   R1, R0, R2                              ; 0849
                           50         0F    AE  9A  001B8          MOVZBL  TIME_BUF+3, R0                          ; 0851
                           52               50  C0  001BC          ADDL2   R0, R2                                  ;
                           52  FFFF2FB0     E2  9E  001BF          MOVAB   -53328(R2), DAYS                        ; 0850
                           51         11    AE  9A  001C6          MOVZBL  TIME_BUF+5, R1                          ; 0852
                           51               0A  C4  001CA          MULL2   #10, R1                                 ;
                           50         12    AE  9A  001CD          MOVZBL  TIME_BUF+6, R0                          ;
                           51               50  C0  001D1          ADDL2   R0, R1                                  ;
                           51       FDF0    C1  9E  001D4          MOVAB   -528(R1), HOURS                         ;
                           50         14    AE  9A  001D9          MOVZBL  TIME_BUF+8, R0                          ; 0853
                           50               0A  C4  001DD          MULL2   #10, R0                                 ;
                           53         15    AE  9A  001E0          MOVZBL  TIME_BUF+9, R3                          ;
                           50               53  C0  001E4          ADDL2   R3, R0                                  ;
                           50       FDF1    C0  9E  001E7          MOVAB   -527(R0), MINUTES                       ;
                           51               3C  C4  001EC          MULL2   #60, R1                                 ; 0857
                           50             7041  9E  001EF          MOVAB   -(MINUTES)[R1], R0                      ;
                           52  000005A0     8F  C4  001F3          MULL2   #1440, R2                               ;
                           50               52  C0  001FA 13$:     ADDL2   R2, R0                                  ;
                           51               50  4E  001FD 14$:     CVTLF   R0, RESULT                              ;
                           50               51  D0  00200          MOVL    RESULT, R0                              ; 0861
                                                04  00203          RET                                             ;
                                            50  D4  00204 15$:     CLRL    R0                                      ; 0884
                                                04  00206          RET                                             ;

; Routine Size:  519 bytes,    Routine Base:  _BAS$CODE + 031C

;  795          0885  1
```

```
;  797          0886  1
;  798          0887  1 END                                ! end of BAS$DATE_TIME
;  799          0888  0 ELUDOM



;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                      1315  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                              Library Statistics
;
;                                        --------  Symbols  --------      Pages       Processing
;       File                             Total     Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776        14         0        581          00:01.0



;                                       COMMAND QUALIFIERS

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

; Size:          1315 code + 0 data bytes
; Run Time:        00:23.1
; Elapsed Time:    00:50.8
; Lines/CPU Min:    2303
; Lexemes/CPU-Min: 22713
; Memory Used:  168 pages
; Compilation Complete
```

BASDET
LIS

BASDISPAT
LIS

BASCVTTP
LIS

BASDELETE
LIS

BASDATETI
LIS

BASECHO
LIS

BASCVTRP
LIS