```
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
```

FILEID**DETACHED

```
DDDDDDDD    EEEEEEEEEE  TTTTTTTTTT    AAAAAA      CCCCCCCC  HH      HH  EEEEEEEEEE  DDDDDDDD
DDDDDDDD    EEEEEEEEEE  TTTTTTTTTT    AAAAAA      CCCCCCCC  HH      HH  EEEEEEEEEE  DDDDDDDD
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD
DD      DD  EEEEEEEE        TT       AA      AA  CC          HHHHHHHHHH  EEEEEEEE    DD      DD
DD      DD  EEEEEEEE        TT       AA      AA  CC          HHHHHHHHHH  EEEEEEEE    DD      DD
DD      DD  EE              TT       AAAAAAAAAA  CC          HH      HH  EE          DD      DD
DD      DD  EE              TT       AAAAAAAAAA  CC          HH      HH  EE          DD      DD
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD      ....
DD      DD  EE              TT       AA      AA  CC          HH      HH  EE          DD      DD      ....
DDDDDDDD    EEEEEEEEEE      TT       AA      AA    CCCCCCCC  HH      HH  EEEEEEEEEE  DDDDDDDD      ....
DDDDDDDD    EEEEEEEEEE      TT       AA      AA    CCCCCCCC  HH      HH  EEEEEEEEEE  DDDDDDDD      ....

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 detached (IDENT = 'V04-000',
;     2          0002  0        ADDRESSING_MODE(EXTERNAL = GENERAL)) =
;     3          0003  1 BEGIN
;     4          0004  1
;     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 ! FACILITY:  Login
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This module handles all processing specific to detached jobs.
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:
;    37          0037  1 !
;    38          0038  1 !       VAX/VMS operating system.
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:  Tim Halvorsen, March 1981
;    41          0041  1 !
;    42          0042  1 ! Modified by:
;    43          0043  1 !
;    44          0044  1 !       V03-013 MHB0147         Mark Bramhall           7-May-1984
;    45          0045  1 !               Guard again no global buffers when opening NETUAF.
;    46          0046  1 !
;    47          0047  1 !       V03-012 MHB0125         Mark Bramhall           10-Apr-1984
;    48          0048  1 !               Set node name, etc. via SET_NODE_NAME.
;    49          0049  1 !               Disallow network access to accounts with secondary passwords.
;    50          0050  1 !               Fix up network output file name scanning.
;    51          0051  1 !
;    52          0052  1 !       V03-011 MHB0107         Mark Bramhall           21-Mar-1984
;    53          0053  1 !               Use LNM services for logical names.
;    54          0054  1 !
;    55          0055  1 !       V03-010 PCG0001         Peter George            31-Jan-1984  15:10
;    56          0056  1 !               Add secondary password to network processing.
;    57          0057  1 !               Correct bug in interpreting batch item list.
```

```
;    58          0058  1 ;
;    59          0059  1 ;       V03-009 ACG0385         Andrew C. Goldstein,    29-Dec-1983  9:59
;    60          0060  1 ;               Implement job type in JIB; fix coding of field references
;    61          0061  1 ;               to proxy file record. Change UAF working set fields
;    62          0062  1 ;               to longwords.
;    63          0063  1 ;
;    64          0064  1 ;       V03-008 ACG0376         Andrew C. Goldstein,    22-Nov-1983  17:16
;    65          0065  1 ;               Interface cleanup with VALIDATE_UAFREC; fix error handling
;    66          0066  1 ;               in GET_PROXY. Put batch input file name in PPD$T_FILENAME.
;    67          0067  1 ;
;    68          0068  1 ;       V03-007 GAS0183         Gerry Smith             16-Sep-1983
;    69          0069  1 ;               For network logins, rearrange the code so that the
;    70          0070  1 ;               node name gets set early on.  This helps in both
;    71          0071  1 ;               accounting and breakin evasion.
;    72          0072  1 ;
;    73          0073  1 ;       V03-006 GAS0164         Gerry Smith             30-Jul-1983
;    74          0074  1 ;               Change the method of disabling logical name translation
;    75          0075  1 ;               in RMS calls to use the new ACMODES field.
;    76          0076  1 ;
;    77          0077  1 ;       V03-005 MLJ0115         Martin L. Jack, 29-Jul-1983  10:29
;    78          0078  1 ;               Update for new log file error handling.
;    79          0079  1 ;
;    80          0080  1 ;       V03-004 GAS0137         Gerry Smith             26-May-1983
;    81          0081  1 ;               Do not signal a $SNDJBC error when terminating a batch job.
;    82          0082  1 ;
;    83          0083  1 ;       V03-003 GAS0123         Gerry Smith             19-Apr-1983
;    84          0084  1 ;               Change interface to use SNDJBC for batch jobs.  Also,
;    85          0085  1 ;               if proxy access is requested and the NETUAF cannot be
;    86          0086  1 ;               accessed, signal with a fatal error.
;    87          0087  1 ;
;    88          0088  1 ;       V03-002 GAS0097         Gerry Smith              4-Jan-1983
;    89          0089  1 ;               Fix the case of proxy login for wildcard entries.
;    90          0090  1 ;
;    91          0091  1 ;       V03-001 GAS0057         Gerry Smith             17-Mar-1982
;    92          0092  1 ;               Fix FABS to disable all but system
;    93          0093  1 ;               logical name translation during open/creates.
;    94          0094  1 ;
;    95          0095  1 ;       V03-010 MLJ34580        Martin L. Jack, 1-Feb-1982  0:55
;    96          0096  1 ;               Make use of extensions to DJT record to set name and /NOTIFY
;    97          0097  1 ;               status for log file print job.  Correct queue name translation
;    98          0098  1 ;               so that explicit queue name is not translated and implicit
;    99          0099  1 ;               SYS$PRINT uses standard queue-name translation modiroutine.
;   100          0100  1 ;
;   101          0101  1 ;       V03-009 GAS0032         Gerry Smith             07-Jan-1982
;   102          0102  1 ;               On proxy login, if no UAF record is found, return
;   103          0103  1 ;               FALSE to indicate lookup failure.
;   104          0104  1 ;
;   105          0105  1 ;       V03-008 GAS0031         Gerry Smith             04-Jan-1982
;   106          0106  1 ;               Remove NETUAF structure definitions from this module.
;   107          0107  1 ;               $NAFDEF now resides in LIB.REQ.
;   108          0108  1 ;
;   109          0109  1 ;       V03-007 SPF0050         Steve Forgey            01-Jan-1982
;   110          0110  1 ;               Store remote node info in P1 space for network jobs.
;   111          0111  1 ;
;   112          0112  1 ;       V03-006 GAS0029         Gerry Smith             31-Dec-1981
;   113          0113  1 ;               Add proxy login for network jobs.
;   114          0114  1 ;
```

```
;  115     0115  1 !        V03-005 HRJ0032         Herb Jacobs             12-Nov-1981
;  116     0116  1 !                Process batch queue WSEXTENT if passed, validate username
;  117     0117  1 !                as valid for batch job, and allow handler to stop a batch
;  118     0118  1 !                job.
;  119     0119  1 !
;  120     0120  1 !        V004    TMH0004         Tim Halvorsen           26-Oct-1981
;  121     0121  1 !                Get ORIGUIC and OUTFNM from LGI area rather than from PPD.
;  122     0122  1 !                Add extra acmode argument to calls to exec_crelog
;  123     0123  1 !                Make use of global SYS$ERROR descriptor, rather than
;  124     0124  1 !                re-translating the logical name again here.
;  125     0125  1 !
;  126     0126  1 !        V003    GWF0073         Gary Fowler             27-Jul-1981
;  127     0127  1 !                Change job name to ASCIC string.  Increase maximum length of
;  128     0128  1 !                message that can be received from the job controller
;  129     0129  1 !
;  130     0130  1 !        V002    TMH0002         Tim Halvorsen           16-Jul-1981
;  131     0131  1 !                Reference SHRLIB$ for shared require files.
;  132     0132  1 !
;  133     0133  1 !        V03-001 GWF0051         Gary W. Fowler          29-May-1981
;  134     0134  1 !                Add file size in message sent when log file is queued for
;  135     0135  1 !                printing.
;  136     0136  1 !--
;  137     0137  1
;  138     0138  1 !
;  139     0139  1 ! Include files
;  140     0140  1 !
;  141     0141  1
;  142     0142  1 LIBRARY 'SYS$LIBRARY:LIB';                  ! VAX/VMS system definitions
;  143     0143  1 REQUIRE 'SHRLIB$:UTILDEF';                  ! Common BLISS definitions
;  144     0328  1
;  145     0329  1 REQUIRE 'LIB$:PPDDEF';                      ! Process permanent data region
;  146     0476  1 REQUIRE 'LIB$:LGIDEF';                      ! LOGINOUT private permanent storage
```

```
;  148          0547  1 !
;  149          0548  1 ! Table of contents
;  150          0549  1 !
;  151          0550  1
;  152          0551  1 FORWARD ROUTINE
;  153          0552  1     init_batch:          NOVALUE,        ! Initialize batch job step
;  154          0553  1     stop_batch_job:      NOVALUE,        ! Stop batch job stream
;  155          0554  1     terminate_batch:     NOVALUE,        ! Stop a batch job
;  156          0555  1     init_network:        NOVALUE,        ! Initialize network job
;  157          0556  1     get_proxy;                           ! Get proxy username
;  158          0557  1
;  159          0558  1 !
;  160          0559  1 ! External routines
;  161          0560  1 !
;  162          0561  1
;  163          0562  1 EXTERNAL ROUTINE
;  164          0563  1     close_output:        NOVALUE,        ! Close primary output file
;  165          0564  1     validate_uafrec:     NOVALUE,        ! Read/validate UAF record
;  166          0565  1     get_uafrec:          NOVALUE,        ! Read UAF record without validiation
;  167          0566  1     logout_message:      NOVALUE,        ! Write logout message
;  168          0567  1     map_imgact:          NOVALUE,        ! Map image activator code segment
;  169          0568  1     create_logical,                      ! Create logical name with LNM services
;  170          0569  1     set_sysprv:          NOVALUE,        ! Turn on SYSPRV
;  171          0570  1     clear_sysprv:        NOVALUE,        ! Turn off SYSPRV
;  172          0571  1     set_uic,                             ! Set process UIC
;  173          0572  1     set_node_name:       NOVALUE,        ! Set remote node info in P1 space
;  174          0573  1     exit_process:        NOVALUE,        ! Exit the process
;  175          0574  1     lib$fid_to_name;                     ! Translate file ID to file name
;  176          0575  1
;  177          0576  1 !
;  178          0577  1 ! Define literals
;  179          0578  1 !
;  180          0579  1
;  181          0580  1
;  182          0581  1 !
;  183          0582  1 ! Define message codes
;  184          0583  1 !
;  185          0584  1
;  186          0585  1 EXTERNAL LITERAL
;  187          0586  1     lgi$_jbcmixup,
;  188          0587  1     lgi$_userauth,
;  189          0588  1     lgi$_netuafacc;
;  190          0589  1
;  191          0590  1 !
;  192          0591  1 ! External storage
;  193          0592  1 !
;  194          0593  1
;  195          0594  1 EXTERNAL
;  196          0595  1     pcb_sts:             BITVECTOR,      ! PCB status flags
;  197          0596  1     job_type,                            ! Job type code for JIB
;  198          0597  1     input_fab:           BBLOCK,         ! Input FAB
;  199          0598  1     input_nam:           BBLOCK,         ! Input NAM
;  200          0599  1     output_fab:          BBLOCK,         ! Output FAB
;  201          0600  1     output_nam:          BBLOCK,         ! Output NAM
;  202          0601  1     uaf_record:          REF BBLOCK,     ! Address of UAF record
;  203          0602  1     sys$input:           VECTOR,         ! Translation of SYS$INPUT
;  204          0603  1     sys$output:          VECTOR,         ! Translation of SYS$OUTPUT
```

```
;  205          0604  1      sys$error:              VECTOR,         ! Translation of SYS$ERROR
;  206          0605  1      ctl$ag_clidata;                         ! Process permanent data storage
;  207          0606  1
;  208          0607  1  BIND
;  209          0608  1      ppd = ctl$ag_clidata: BBLOCK;           ! Address of PPD structure
;  210          0609  1
;  211          0610  1
```

```
;  213          0611  1 GLOBAL ROUTINE init_batch: NOVALUE =
;  214          0612  1
;  215          0613  1 !---
;  216          0614  1 !
;  217          0615  1 !       Perform batch initialization by requesting the job parameters
;  218          0616  1 !       from the job controller.
;  219          0617  1 !
;  220          0618  1 ! Inputs:
;  221          0619  1 !
;  222          0620  1 !       None
;  223          0621  1 !
;  224          0622  1 ! Outputs:
;  225          0623  1 !
;  226          0624  1 !       uaf_record = Address of UAF record for user
;  227          0625  1 !---
;  228          0626  1
;  229          0627  2 BEGIN
;  230          0628  2
;  231          0629  2 OWN
;  232          0630  2     jobname:    VECTOR [43,BYTE],       ! Must be static to be passed back
;  233          0631  2     logfile:    VECTOR [nam$c_maxrss,BYTE];     ! to caller as output filespec
;  234          0632  2
;  235          0633  2 LOCAL
;  236          0634  2     username : VECTOR[2]                ! Descriptor for username
;  237          0635  2                INITIAL(REP 2 OF (0)),
;  238          0636  2     logdesc:    VECTOR [2],             ! Logical name descriptor
;  239          0637  2     logname:    VECTOR [2,BYTE],        ! 2 character logical name
;  240          0638  2     ptr : REF VECTOR[,WORD],
;  241          0639  2     length,
;  242          0640  2     buffer : VECTOR[500],               ! SNDJBC message buffer
;  243          0641  2     flags : REF $BBLOCK INITIAL (0);    ! Flags from job controller
;  244          0642  2
;  245          0643  2 !
;  246          0644  2 ! Check to see if at early termination of batch job.
;  247          0645  2 !
;  248          0646  2 IF .ppd [ppd$w_outifi] NEQ 0            ! If not first job step,
;  249          0647  2     AND NOT .ppd [ppd$l_lststatus]      ! and job step failed,
;  250          0648  3     AND ((.ppd [ppd$l_lststatus] AND 6) NEQ 0) ! and its an error or fatal,
;  251          0649  2 THEN
;  252          0650  2     terminate_batch(0);         ! Stop the batch job
;  253          0651  2
;  254          0652  2 !
;  255          0653  2 ! Request detached job step initialization parameters from job controller
;  256          0654  2 !
;  257          0655  3 BEGIN
;  258          0656  3 LOCAL
;  259          0657  3     status,
;  260          0658  3     iosb : VECTOR[2],
;  261          0659  3     itmlst : $ITMLST_DECL(ITEMS = 1);
;  262          0660  3
;  263        P 0661  3 $ITMLST_INIT(ITMLST = itmlst,
;  264        P 0662  3                 (ITMCOD = sjc$_batch_output,
;  265        P 0663  3                  BUFSIZ = %ALLOCATION(buffer),
;  266          0664  3                  BUFADR = buffer));
;  267        P 0665  3 status = $SNDJBCW(FUNC   = sjc$_batch_service,
;  268        P 0666  3                   ITMLST = itmlst,
;  269          0667  3                   IOSB   = iosb);
```

```
;  270     0668  3 IF .status
;  271     0669  3 THEN status = .iosb[0];
;  272     0670  3 IF NOT .status
;  273     0671  3 THEN SIGNAL_STOP(.status);
;  274     0672  2 END;
;  275     0673  2
;  276     0674  2 !
;  277     0675  2 ! Find the flags longword and the username.
;  278     0676  2 !
;  279     0677  2 ptr = buffer;
;  280     0678  2 WHILE true DO
;  281     0679  3     BEGIN
;  282     0680  3     IF .ptr[1] EQL 0
;  283     0681  3     THEN EXITLOOP;
;  284     0682  3     IF .ptr[1] EQL dji$k_flags
;  285     0683  3     THEN
;  286     0684  4         BEGIN
;  287     0685  4         flags = ptr[2];
;  288     0686  4         IF .flags[dji$v_terminate]
;  289     0687  4         THEN stop_batch_job(.flags, buffer, 0);
;  290     0688  4         IF .flags[dji$v_delete_file]
;  291     0689  4         THEN input_fab[fab$v_dl[t] = true;
;  292     0690  4         IF .flags[dji$v_restarting]
;  293     0691  4         THEN ppd[ppd$v_restart] = true;
;  294     0692  4         IF .username[1] NEQ 0
;  295     0693  4         OR .ppd[ppd$w_outif'] NEQ 0
;  296     0694  4         THEN EXITLOOP;
;  297     0695  4         END
;  298     0696  3     ELSE IF .ptr[1] EQL dji$k_username
;  299     0697  3     THEN
;  300     0698  4         BEGIN
;  301     0699  4         username[1] = ptr[2];
;  302     0700  4         IF .flags NEQ 0
;  303     0701  4         THEN EXITLOOP;
;  304     0702  3         END;
;  305     0703  3     ptr = ptr[2] + .ptr[0];
;  306     0704  2     END;
;  307     0705  2
;  308     0706  2 !
;  309     0707  2 ! If this is the first job step, then do first_time_thru stuff.
;  310     0708  2 !
;  311     0709  2 IF .ppd [ppd$w_outifi] EQL 0            ! If this the first job step,
;  312     0710  2 THEN
;  313     0711  3     BEGIN
;  314     0712  3     job_type = jib$c_batch;
;  315     0713  3 !***username [0] = uaf$s_username;      ! Setup descriptor of user name
;  316     0714  3     username [0] = 12;                  ! Setup descriptor of user name
;  317     0715  3     get_uafrec(username);               ! Get user's UAF record
;  318     0716  3     IF .uaf_record EQL 0
;  319     0717  3     THEN
;  320     0718  3         SIGNAL_STOP(lgi$_userauth);     ! signal fatal error
;  321     0719  2     END;
;  322     0720  2
;  323     0721  2 !
;  324     0722  2 ! Now to go thru all the data items in BUFFER, setting up the input and
;  325     0723  2 ! output files as indicated, as well as working set parameters and cpu
;  326     0724  2 ! time limit, if first job step.
```

```
;  327     0725  2 !
;  328     0726  2 logdesc[0] = 2;                         ! Set up the logical name descriptor
;  329     0727  2 logdesc[1] = logname;
;  330     0728  2 logname[0] = 'P';
;  331     0729  2
;  332     0730  2 output_fab[fab$b_fns] = 0;              ! Initialize the file name
;  333     0731  2 ptr = buffer;
;  334     0732  2 WHILE true DO
;  335     0733  3     BEGIN
;  336     0734  3     CASE .ptr[1] FROM 0 TO dji$k_wsquota OF
;  337     0735  3         SET
;  338     0736  3         [0] : EXITLOOP;
;  339     0737  3
;  340     0738  3         [dji$k_wsdefault] :
;  341     0739  3             IF .ppd[ppd$w_outifi] EQL 0         ! If first job step, set wsdefault
;  342     0740  3             THEN
;  343     0741  4                 BEGIN
;  344     0742  4                 IF .flags[dji$v_use_wsdefault]
;  345     0743  5                 THEN uaf_record[uaf$l_dfwscnt] = .(ptr[2])
;  346     0744  4                 ELSE uaf_record[uaf$l_dfwscnt] = MINU(.(ptr[2]),
;  347     0745  4                                                       .uaf_record[uaf$l_dfwscnt]);
;  348     0746  3                 END;
;  349     0747  3
;  350     0748  3         [dji$k_wsextent] :
;  351     0749  3             IF .ppd[ppd$w_outifi] EQL 0         ! If first job step, set wsextent
;  352     0750  3             THEN
;  353     0751  4                 BEGIN
;  354     0752  4                 IF .flags[dji$v_use_wsextent]
;  355     0753  5                 THEN uaf_record[uaf$l_wsextent] = .(ptr[2])
;  356     0754  4                 ELSE uaf_record[uaf$l_wsextent] = MINU(.(ptr[2]),
;  357     0755  4                                                        .uaf_record[uaf$l_wsextent]);
;  358     0756  3                 END;
;  359     0757  3
;  360     0758  3         [dji$k_wsquota] :
;  361     0759  3             IF .ppd[ppd$w_outifi] EQL 0         ! If first job step, set wsquota
;  362     0760  3             THEN
;  363     0761  4                 BEGIN
;  364     0762  4                 IF .flags[dji$v_use_wsquota]
;  365     0763  5                 THEN uaf_record[uaf$l_wsquota] = .(ptr[2])
;  366     0764  4                 ELSE uaf_record[uaf$l_wsquota] = MINU(.(ptr[2]),
;  367     0765  4                                                       .uaf_record[uaf$l_wsquota]);
;  368     0766  3                 END;
;  369     0767  3
;  370     0768  3         [dji$k_cpu_maximum] :
;  371     0769  3             IF .ppd[ppd$w_outifi] EQL 0         ! If first job step, set CPU time limit
;  372     0770  3             THEN
;  373     0771  4                 BEGIN
;  374     0772  4                 IF .flags[dji$v_use_cpu_maximum]
;  375     0773  5                 THEN uaf_record[uaf$l_cputim] = .(ptr[2])
;  376     0774  4                 ELSE uaf_record[uaf$l_cputim] = MINU(.(ptr[2])-1,       ! So that 0 > all others
;  377     0775  4                                                      .uaf_record[uaf$l_cputim]-1) + 1;
;  378     0776  3                 END;
;  379     0777  3
;  380     0778  3         [dji$k_job_name] :
;  381     0779  4             BEGIN                           ! Setup output log file name from job name
;  382     0780  4             length = .ptr[0];               ! get length of job name
;  383     0781  4             CH$MOVE(.length,
```

```
;  384     0782  4                    ptr[2],
;  385     0783  4                    jobname);
;  386     0784  4            CH$MOVE(4, UPLIT BYTE('.LOG'), jobname + .length);
;  387     0785  4            output_fab [fab$l_dna] = jobname;   ! Set default to <jobname>.LOG
;  388     0786  4            output_fab [fab$b_dns] = .length + 4;
;  389     0787  3            END;
;  390     0788  3
;  391     0789  3        [dji$k_log_specification] :
;  392     0790  4            BEGIN                   ! Set up the log file name
;  393     0791  4            output_fab[fab$b_fns] = .ptr[0];
;  394     0792  4            output_fab[fab$l_fna] = logfile;
;  395     0793  4            CH$MOVE(.ptr[0],
;  396     0794  4                ptr[2],
;  397     0795  4                logfile);
;  398     0796  3            END;
;  399     0797  3
;  400     0798  3        [dji$k_file_identification] :   ! Batch input file
;  401     0799  4            BEGIN
;  402     0800  4            LOCAL
;  403     0801  4                name_desc       : VECTOR [2],
;  404     0802  4                dvi_desc        : VECTOR [2],
;  405     0803  4                name_length     : WORD;
;  406     0804  4            CH$MOVE(ppd$c_dvifid,
;  407     0805  4                    ptr[2],
;  408     0806  4                    input_nam[nam$t_dvi]);
;  409     0807  4            input_fab [fab$v_nam] = true; ! Mark to open input by NAM block
;  410     0808  4                                          ! Get input file name for CLI
;  411     0809  4            name_desc[0] = ppd$s_filename-1;
;  412     0810  4            name_desc[1] = ppd[ppd$t_filename]+1;
;  413     0811  4            dvi_desc[0] = .VECTOR [input_nam[nam$t_dvi], 0; ,BYTE];
;  414     0812  4            dvi_desc[1] = input_nam[nam$t_dvi]+1;
;  415     0813  4            IF [ib$fid_to_name (dvi_desc, input_nam[nam$w_fid],
;  416     0814  4                                name_desc, name_length)
;  417     0815  4            THEN VECTOR [ppd[ppd$t_filename], 0; ,BYTE] = .name_length;
;  418     0816  3            END;
;  419     0817  3
;  420     0818  3        [dji$k_parameter_1 TO dji$k_parameter_8] :
;  421     0819  4            BEGIN
;  422     0820  4            LOCAL
;  423     0821  4                desc: VECTOR[2];
;  424     0822  4            desc[0] = .ptr[0];
;  425     0823  4            desc[1] = ptr[2];
;  426     0824  4            logname [1] = '1' + .ptr[1] - dji$k_parameter_1;
;  427     0825  4            create_logical(logdesc,             ! Create Pn logical name
;  428     0826  4                           desc,
;  429     0827  4                           psl$c_user);
;  430     0828  3            END;
;  431     0829  3
;  432     0830  3        [dji$k_restart] :
;  433     0831  4            BEGIN
;  434     0832  4            LOCAL
;  435     0833  4                desc : VECTOR[2];
;  436     0834  4            desc[0] = .ptr[0];
;  437     0835  4            desc[1] = ptr[2];
;  438     0836  4            create_logical(%ASCID 'BATCH$RESTART',
;  439     0837  4                           desc,
;  440     0838  4                           psl$c_user);
```

```
;  441          0839  3              END;
;  442          0840  3
;  443          0841  3          [INRANGE] : true;
;  444          0842  3          [OUTRANGE] : true;
;  445          0843  3          TES;
;  446          0844  3      ptr = ptr[2] + .ptr[0];
;  447          0845  2      END;
;  448          0846  2
;  449          0847  2  IF .flags[dji$v_log_null]
;  450          0848  2  THEN
;  451          0849  3      BEGIN
;  452          0850  3      output_fab [fab$b_fns] = 4;
;  453          0851  3      output_fab [fab$l_fna] = UPLIT BYTE('_NL:');
;  454          0852  2      END;
;  455          0853  2
;  456          0854  2
;  457          0855  1 END;


                                                                .TITLE  DETACHED
                                                              . .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                 47  4F  4C  2E  00000 P.AAA:  .ASCII  \.LOG\
00  00  54  52  41  54  53  45  52  24  48  43  54  41  42  00  00004 P.AAC:  .ASCII  \BATCH$RESTART\<0><0><0>                ;
                                                                 00013                                                      ;
                                                             010E000D  00014 P.AAB:  .LONG   17694733                       ;
                                                             00000000' 00018        .ADDRESS P.AAC                      ;
                                                 3A  4C  4E  5F  0001C P.AAD:  .ASCII  \_NL:\                            ;

                                                                .PSECT  $OWN$,NOEXE,2

                                                        00000 JOBNAME:.BLKB   43
                                                        0002B         .BLKB   1
                                                        0002C LOGFILE:.BLKB   255

                                                                .EXTRN  CLOSE_OUTPUT, VALIDATE_UAFREC
                                                                .EXTRN  GET_UAFREC, LOGOUT_MESSAGE
                                                                .EXTRN  MAP_IMGACT, CREATE_LOGICAL
                                                                .EXTRN  SET_SYSPRV, CLEAR_SYSPRV
                                                                .EXTRN  SET_UIC, SET_NODE_NAME
                                                                .EXTRN  EXIT_PROCESS, LIB$FID_TO_NAME
                                                                .EXTRN  LGI$_JBCMIXUP, LGI$_USERAUTH
                                                                .EXTRN  LGI$_NETUAFACC, PCB_STS
                                                                .EXTRN  JOB_TYPE, INPUT_FAB
                                                                .EXTRN  INPUT_NAM, OUTPUT_FAB
                                                                .EXTRN  OUTPUT_NAM, UAF_RECORD
                                                                .EXTRN  SYS$INPUT, SYS$OUTPUT
                                                                .EXTRN  SYS$ERROR, CTL$AG_CLIDATA
                                                                .EXTRN  SYS$SNDJBCW

                                                                .PSECT  $CODE$,NOWRT,2

                                                    0FFC 00000          .ENTRY  INIT_BATCH, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 0611
                                                                                R10,R11                                         ;
                                         5B 00000000G   00   9E 00002          MOVAB   UAF_RECORD, R11                        ;
```

```
                    5A 00000000G   00  9E 00009          MOVAB   OUTPUT_FAB+52, R10                ;
                    59 00000000G   00  9E 00010          MOVAB   PPD+36, R9                      ;
                    5E       F800  CE  9E 00017          MOVAB   -2048(SP), SP                   ;
                             F8    AD  7C 0001C          CLRQ    USERNAME                        ; 0627
                                   57  D4 0001F          CLRL    FLAGS                           ;
                                   69  B5 00021          TSTW    PPD+36                          ; 0646
                                   11  13 00023          BEQL    1$                              ;
                    0D         F4  A9  E8 00025          BLBS    PPD+24, 1$                      ; 0647
                    06         F4  A9  93 00029          BITB    PPD+24, #6                      ; 0648
                                   07  13 0002D          BEQL    1$                              ;
                                   7E  D4 0002F          CLRL    -(SP)                           ; 0650
                0000V  CF          01  FB 00031          CALLS   #1, TERMINATE_BATCH             ;
                    50         08  AE  9E 00036 1$:      MOVAB   ITMLST, $$ITMBLKPTR             ; 0664
                    80 000B07D0    8F  D0 0003A          MOVL    #722896, ($$ITMBLKPTR)+         ;
                    80         20  AE  9E 00041          MOVAB   BUFFER, ($$ITMBLKPTR)+          ;
                                   80  7C 00045          CLRQ    ($$ITMBLKPTR)+                  ;
                                   7E  7C 00047          CLRQ    -(SP)                           ; 0667
                                20  AE  9F 00049          PUSHAB  IOSB                            ;
                                14  AE  9F 0004C          PUSHAB  ITMLST                          ;
                    7E                 07  7D 0004F          MOVQ    #7, -(SP)                       ;
                                       7E  D4 00052          CLRL    -(SP)                           ;
            00000000G  00              07  FB 00054          CALLS   #7, SYS$SNDJBCW                 ;
                    07                 50  E9 0005B          BLBC    STATUS, 2$                      ; 0668
                    50             18  AE  D0 0005E          MOVL    IOSB, STATUS                    ; 0669
                    09                 50  E8 00062          BLBS    STATUS, 3$                      ; 0670
                                       50  DD 00065 2$:      PUSHL   STATUS                          ; 0671
            00000000G  00              01  FB 00067          CALLS   #1, LIB$STOP                    ;
                    56             20  AE  9E 0006E 3$:      MOVAB   BUFFER, PTR                     ; 0677
                    50             02  A6  3C 00072 4$:      MOVZWL  2(PTR), R0                      ; 0680
                                       4D  13 00076          BEQL    11$                             ;
                                   03  50  B1 00078          CMPW    R0, #3                          ; 0682
                                       30  12 0007B          BNEQ    8$                              ;
                                   57  04  A6  9E 0007D      MOVAB   4(R6), FLAGS                    ; 0685
                0C                 67      06  E1 00081      BBC     #6, (FLAGS), 5$                 ; 0686
                                           7E  D4 00085      CLRL    -(SP)                           ; 0687
                                       24  AE  9F 00087      PUSHAB  BUFFER                          ;
                                           57  DD 0008A      PUSHL   FLAGS                           ;
                        0000V      CF      03  FB 0008C      CALLS   #3, STOP_BATCH_JOB              ;
                                   08      67  E9 00091 5$:  BLBC    (FLAGS), 6$                     ; 0688
                    00000000G      00  80  8F  88 00094      BISB2   #128, INPUT_FAB+5               ; 0689
                04                 67      05  E1 0009C 6$:  BBC     #5, (FLAGS), 7$                 ; 0690
                                DE     A9      10  88 000A0      BISB2   #16, PPD+2                      ; 0691
                                       FC  AD  D5 000A4 7$:      TSTL    USERNAME+4                      ; 0692
                                           1C  12 000A7          BNEQ    11$                             ;
                                           69  B5 000A9          TSTW    PPD+36                          ; 0693
                                           0C  11 000AB          BRB     9$                              ;
                                   10          50  B1 000AD 8$:  CMPW    R0, #16                         ; 0696
                                               09  12 000B0      BNEQ    10$                             ;
                                FC     AD      04  A6  9E 000B2  MOVAB   4(R6), USERNAME+4               ; 0699
                                               57  D5 000B7      TSTL    FLAGS                           ; 0700
                                               0A  12 000B9 9$:  BNEQ    11$                             ;
                                   50          66  3C 000BB 10$: MOVZWL  (PTR), R0                      ; 0703
                                   56      04 A046 9E 000BE      MOVAB   4(R0)[PTR], PTR                 ;
                                               AD  11 000C3      BRB     4$                              ; 0678
                                               69  B5 000C5 11$: TSTW    PPD+36                          ; 0709
                                               26  12 000C7      BNEQ    12$                             ;
                    00000000G  00              02  D0 000C9      MOVL    #2, JOB_TYPE                    ; 0712
```

```
                                F8   AD                  0C   D0 000D0          MOVL    #12, USERNAME                                ; 0714
                                                   F8   AD   9F 000D4          PUSHAB  USERNAME                                     ; 0715
                          00000000G  00                  01   FB 000D7          CALLS   #1, GET_UAFREC
                                                         6B   D5 000DE          TSTL    UAF_RECORD                                   ; 0716
                                                         0D   12 000E0          BNEQ    12$
                                     00000000G           8F   DD 000E2          PUSHL   #LGI$_USERAUTH                               ; 0718
                          00000000G  00                  01   FB 000E8          CALLS   #1, LIB$STOP
                                F0   AD                  02   D0 000EF 12$:     MOVL    #2, LOGDESC                                  ; 0726
                                F4   AD             04   AE   9E 000F3          MOVAB   LOGNAME, LOGDESC+4                           ; 0727
                                04   AE             50   8F   90 000F8          MOVB    #80, LOGNAME                                 ; 0728
                                                         6A   94 000FD          CLRB    OUTPUT_FAB+52                                ; 0730
                                     56             20   AE   9E 000FF          MOVAB   BUFFER, PTR                                  ; 0731
                              13     00             02   A6   AF 00103 13$:     CASEW   2(PTR), #0, #19                              ; 0734
                        0158          007E              0163      00108 14$:     .WORD   37$-14$,-
                        0126          0158              00AB      00110                    23$-14$,-
                        0126          0126              0126      00118                    31$-14$,-
                        013F          0126              0126      00120                    36$-14$,-
                        0057          002A              0158      00128                    27$-14$,-
                                                                                           36$-14$,-
                                                                                           29$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           33$-14$,-
                                                                                           34$-14$,-
                                                                                           36$-14$,-
                                                                                           15$-14$,-
                                                                                           17$-14$,-
                                                                                           18$-14$
                                                         7F   11 00130          BRB     26$
                                                         69   B5 00132 15$:     TSTW    PPD+36                                       ; 0739
                                                         7B   12 00134          BNEQ    26$
                                50                       6B   D0 00136          MOVL    UAF_RECORD, R0                               ; 0743
                                51             0220      C0   9E 00139          MOVAB   544(R0), R1
                                2D               01      A7   E8 0013E          BLBS    1(FLAGS), 19$                                ; 0742
                                50               04      A6   D0 00142 16$:     MOVL    4(PTR), R0                                   ; 0745
                                61                       50   D1 00146          CMPL    R0, (R1)
                                                         33   1A 00149          BGTRU   21$
                                                         34   11 0014B          BRB     22$                                          ; 0744
                                                         69   B5 0014D 17$:     TSTW    PPD+36                                       ; 0749
                                                         60   12 0014F          BNEQ    26$
                                50                       6B   D0 00151          MOVL    UAF_RECORD, R0                               ; 0753
                                51             0224      C0   9E 00154          MOVAB   548(R0), R1
                          12    67                       09   E0 00159          BBS     #9, (FLAGS), 19$                             ; 0752
                                                         E3   11 0015D          BRB     16$                                          ; 0755
                                                         69   B5 0015F 18$:     TSTW    PPD+36                                       ; 0759
                                                         6F   12 00161          BNEQ    28$
                                50                       6B   D0 00163          MOVL    UAF_RECORD, R0                               ; 0763
                                51             021C      C0   9E 00166          MOVAB   540(R0), R1
                          06    67                       0A   E1 0016B          BBC     #10, (FLAGS), 20$                            ; 0762
                                61               04      A6   D0 0016F 19$:     MOVL    4(PTR), (R1)                                 ; 0763
                                                         6F   11 00173          BRB     30$
                                50               04      A6   D0 00175 20$:     MOVL    4(PTR), R0                                   ; 0765
```

```
                                61              50  D1 00179          CMPL    R0, (R1)                                        ;
                                                03  1B 0017C          BLEQU   22$                                             ;
                                50              61  D0 C017E 21$:     MOVL    (R1), R0                                        ;
                                61              50  D0 00181 22$:     MOVL    R0, (R1)                                        ; 0764
                                                5E  11 00184          BRB     30$                                             ; 0759
                                                69  B5 00186 23$:     TSTW    PPD+36                                          ; 0769
                                                5A  12 00188          BNEQ    30$                                             ;
                                50              6B  D0 0018A          MOVL    UAF_RECORD, R0                                  ; 0773
                                50      022C    C0  9E 0018D          MOVAB   556(R0), R0                                     ;
                                                67  95 00192          TSTB    (FLAGS)                                         ; 0772
                                                06  18 00194          BGEQ    24$                                             ;
                                60        04    A6  D0 00196          MOVL    4(PTR), (R0)                                    ; 0773
                                                48  11 0019A          BRB     30$                                             ;
                        52      04      A6          01  C3 0019C 24$:     SUBL3   #1, 4(PTR), R2                                  ; 0774
                        51              60          01  C3 001A1          SUBL3   #1, (R0), R1                                    ; 0775
                                        51          52  D1 001A5          CMPL    R2, R1                                          ;
                                                    03  1B 001A8          BLEQU   25$                                             ;
                                        52          51  D0 001AA          MOVL    R1, R2                                          ;
                                        60    01    A2  9E 001AD 25$:     MOVAB   1(R2), (R0)                                     ;
                                                    79  11 001B1 26$:     BRB     32$                                             ; 0769
                                        58          66  3C 001B3 27$:     MOVZWL  (PTR), LENGTH                                   ; 0780
                0000'   CF      04      A6          58  28 001B6          MOVC3   LENGTH, 4(PTR), JOBNAME                         ; 0782
                                        0000'CF     48  9F 001BD          PUSHAB  JOBNAME[LENGTH]                                 ; 0784
                                        9E  0000'   CF  D0 001C2          MOVL    P.AAA, @(SP)+                                   ;
                                FC      AA  0000'   CF  9E 001C7          MOVAB   JOBNAME, OUTPUT_FAB+48                          ; 0785
                  01    AA                  58          04  81 001CD          ADDB3   #4, LENGTH, OUTPUT_FAB+53                       ; 0786
                                                    58  11 001D2 28$:     BRB     32$                                             ; 0734
                                        6A          66  90 001D4 29$:     MOVB    (PTR), OUTPUT_FAB+52                            ; 0791
                                F8      AA  0000'   CF  9E 001D7          MOVAB   LOGFILE, OUTPUT_FAB+44                          ; 0792
                0000'   CF      04      A6          66  28 001DD          MOVC3   (PTR), 4(PTR), LOGFILE                          ; 0794
                                                    7A  11 001E4 30$:     BRB     36$                                             ; 0734
        00000000G       00      04      A6          1C  28 001E6 31$:     MOVC3   #28, 4(PTR), INPUT_NAM+20                       ; 0806
                        00000000G       00          01  88 001EF          BISB2   #1, INPUT_FAB+7                                 ; 0807
                                18      AE      FF  8F  9A 001F6          MOVZBL  #255, NAME_DESC                                 ; 0609
                                1C      AE      45  A9  9E 001FB          MOVAB   PPD+105, NAME_DESC+4                            ; 0810
                                10      AE  00000000G   00  9A 00200          MOVZBL  INPUT_NAM+20, DVI_DESC                          ; 0811
                                14      AE  00000000G   00  9E 00208          MOVAB   INPUT_NAM+21, DVI_DESC+4                        ; 0812
                                                    5E  DD 00210          PUSHL   SP                                              ; 0813
                                                1C  AE  9F 00212          PUSHAB  NAME_DESC                                       ;
                                            00000000G   00  9F 00215          PUSHAB  INPUT_NAM+36                                    ;
                                                1C  AE  9F 0021B          PUSHAB  DVI_DESC                                        ;
                        00000000G       00          04  FB 0021E          CALLS   #4, LIB$FID_TO_NAME                             ;
                                                    38  50  E9 00225          BLBC    R0, 36$                                         ;
                                44      A9          6E  90 00228          MOVB    NAME_LENGTH, PPD+104                            ; 0815
                                                    32  11 0022C 32$:     BRB     36$                                             ; 0734
                                18      AE          66  3C 0022E 33$:     MOVZWL  (PTR), DESC                                     ; 0822
                                1C      AE    04    A6  9E 00232          MOVAB   4(R6), DESC+4                                   ; 0823
                  05    AE      02      A6          2A  81 00237          ADDB3   #42, 2(PTR), LOGNAME+1                          ; 0824
                                                    03  DD 0023D          PUSHL   #3                                              ; 0825
                                                1C  AE  9F 0023F          PUSHAB  DESC                                            ;
                                                F0  AD  9F 00242          PUSHAB  LOGDESC                                         ;
                                                    12  11 00245          BRB     35$                                             ;
                                18      AE          66  3C 00247 34$:     MOVZWL  (PTR), DESC                                     ; 0834
                                1C      AE    04    A6  9E 0024B          MOVAB   4(R6), DESC+4                                   ; 0835
                                                    03  DD 00250          PUSHL   #3                                              ; 0836
                                                1C  AE  9F 00252          PUSHAB  DESC                                            ;
                                            0000'   CF  9F 00255          PUSHAB  P.AAB                                           ;
```

```
                00000000G  00              03  FB 00259 35$:   CALLS   #3, CREATE_LOGICAL                    ;
                           50              66  3C 00260 36$:   MOVZWL  (PTR), R0                           ; 0844
                           56        04 A046  9E 00263         MOVAB   4(R0)[PTR], PTR                     ;
                                        FE98  31 00268         BRW     13$                                 ; 0732
                     09                 67      02  E1 0026B 37$:   BBC     #2, (FLAGS), 38$                ; 0847
                                        6A      04  90 0026F         MOVB    #4, OUTPUT_FAB+52               ; 0850
                              F8        AA  0000'   CF  9E 00272         MOVAB   P.AAD, OUTPUT_FAB+44        ; 0851
                                                    04  00278 38$:   RET                                     ; 0855

; Routine Size:  633 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  459     0856  1 GLOBAL ROUTINE terminate_batch(signal_args : REF $BBLOCK): NOVALUE =
;  460     0857  1
;  461     0858  1 !---
;  462     0859  1 !
;  463     0860  1 !       Request a job controller termination message, then stop batch job.
;  464     0861  1 !
;  465     0862  1 ! Inputs:
;  466     0863  1 !       Signal arguments or 0.
;  467     0864  1 !
;  468     0865  1 ! Outputs:
;  469     0866  1 !
;  470     0867  1 !       Job termination, no return, exit via exit process
;  471     0868  1 !---
;  472     0869  1
;  473     0870  2 BEGIN
;  474     0871  2
;  475     0872  2 LOCAL
;  476     0873  2     status,                             ! Status return from SNDJBC
;  477     0874  2     input_buffer:       $BBLOCK[50],    ! SNDJBC input buffer
;  478     0875  2     buffer:             $BBLOCK[500],   ! SNDJBC output buffer
;  479     0876  2     p : REF $BBLOCK,                    ! Cursor for input buffer
;  480     0877  2     ptr : REF VECTOR[,WORD],            ! Pointer to buffer contents
;  481     0878  2     iosb : VECTOR[2],                   ! Final status from SNDJBC
;  482     0879  2     itmlst : $ITMLST_DECL(ITEMS = 2);   ! SNDJBC item list
;  483     0880  2
;  484     0881  2
;  485     0882  2 !
;  486     0883  2 ! Initialize SNDJBC input buffer.
;  487     0884  2 !
;  488     0885  2 p = input_buffer;
;  489     0886  2
;  490     0887  2 p[dji$w_item_code] = dji$k_input_flags; ! Inhibit return of a file
;  491     0888  2 p[dji$w_item_size] = 4;
;  492     0889  2 p = .p + dji$s_item_header;
;  493     0890  2 .p = dji$m_no_file;
;  494     0891  2 p = .p + 4;
;  495     0892  2
;  496     0893  2 IF .signal_args NEQA 0                  ! If signal arguments present
;  497     0894  2 THEN
;  498     0895  3     BEGIN
;  499     0896  3     LOCAL
;  500     0897  3         i:              REF $BBLOCK;    ! Pointer to item header
;  501     0898  3
;  502     0899  3     i = .p;
;  503     0900  3     p[dji$w_item_code] = dji$k_condition_vector;
;  504     0901  3     p[dji$w_item_size] = 4;
;  505     0902  3     p = .p + dji$s_item_header;
;  506     0903  3     .p = .signal_args[4,0,32,0];        ! Primary condition
;  507     0904  3     p = .p + 4;
;  508     0905  3     IF .signal_args[0,0,8,0] GEQU 3
;  509     0906  3     THEN
;  510     0907  4         BEGIN
;  511     0908  4         i[dji$w_item_size] = 8;
;  512     0909  4         .p = .signal_args[12,0,32,0];   ! STS, if present
;  513     0910  4         p = .p + 4;
;  514     0911  3         END;
;  515     0912  3     IF .signal_args[0,0,8,0] GEQU 4
```

```
;  516          0913  3     THEN
;  517          0914  4         BEGIN
;  518          0915  4         i[dji$w_item_size] = 12;
;  519          0916  4         .p = .signal_args[16,0,32,0];   ! STV, if present
;  520          0917  4         p = .p + 4;
;  521          0918  3         END;
;  522          0919  2     END;
;  523          0920  2 .p = 0;                                 ! Zero terminate list
;  524          0921  2
;  525          0922  2
;  526          0923  2 !
;  527          0924  2 ! Request parameters from job controller.
;  528          0925  2 !
;  529        P 0926  2 $ITMLST_INIT(ITMLST = itmlst,
;  530        P 0927  2                 (ITMCOD = sjc$_batch_input,
;  531        P 0928  2                  BUFSIZ = %ALLOCATION(input_buffer),
;  532        P 0929  2                  BUFADR = input_buffer),
;  533        P 0930  2                 (ITMCOD = sjc$_batch_output,
;  534        P 0931  2                  BUFSIZ = %ALLOCATION(buffer),
;  535          0932  2                  BUFADR = buffer));
;  536        P 0933  2 status = $SNDJBCW(FUNC   = sjc$_batch_service,                                    ;
;  537        P 0934  2                   ITMLST = itmlst,
;  538          0935  2                   IOSB   = iosb);
;  539          0936  2 IF .status
;  540          0937  2 THEN status = .iosb[0];
;  541          0938  2 IF NOT .status
;  542          0939  2 THEN stop_batch_job(UPLIT(0), 0, .signal_args);
;  543          0940  2
;  544          0941  2 !
;  545          0942  2 ! Look for the flags word.  Once that is found, we can call the routine
;  546          0943  2 ! to actually stop this job.
;  547          0944  2 !
;  548          0945  2 ptr = buffer;
;  549          0946  2 WHILE .(ptr[0]) NEQ 0 DO
;  550          0947  3     BEGIN
;  551          0948  3     IF .ptr[1] EQL dji$k_flags
;  552          0949  3     THEN stop_batch_job(ptr[2], buffer, .signal_args);
;  553          0950  3     ptr = ptr[2] + .ptr[0];
;  554          0951  2     END;
;  555          0952  2
;  556          0953  2
;  557          0954  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                            00000000  0000 P.AAE:  .LONG   0                                              ;

                                                .PSECT  $CODE$,NOWRT,2

                                      000C 00000        .ENTRY  TERMINATE_BATCH, Save R2,R3               ; 0856
                    5E      FDB4    CE   9E 00002       MOVAB   -588(SP), SP                              ;
                    50        CC    AD   9E 000C7       MOVAB   INPUT_BUFFER, P                           ; 0885
                    80 80010004     8F   D0 0000B       MOVL    #-2147418108, (P)+                        ; 0888
```

```
                80              01   D0 00012          MOVL    #1, (P)+                        ; 0890
                53        04    AC   D0 00015          MOVL    SIGNAL_ARGS, R3                 ; 0893
                                26   13 00019          BEQL    2$
                51              50   D0 0001B          MOVL    P, I                            ; 0899
                80 80020004     8F   D0 0001E          MOVL    #-2147352572, (P)+              ; 0901
                80        04    A3   D0 00025          MOVL    4(R3), (P)+                     ; 0903
                03              63   91 00029          CMPB    (R3), #3                        ; 0905
                                07   1F 0002C          BLSSU   1$
                61              08   B0 0002E          MOVW    #8, (I)                         ; 0908
                80        0C    A3   D0 00031          MOVL    12(R3), (P)+                    ; 0909
                04              63   91 00035 1$:      CMPB    (R3), #4                        ; 0912
                                07   1F 00038          BLSSU   2$
                61              0C   B0 0003A          MOVW    #12, (I)                        ; 0915
                80        10    A3   D0 0003D          MOVL    16(R3), (P)+                    ; 0916
                                60   D4 00041 2$:      CLRL    (P)                             ; 0920
                50              6E   9E 00043          MOVAB   ITMLST, $$ITMBLKPTR             ; 0932
                80 000A0032     8F   D0 00046          MOVL    #655410, ($$ITMBLKPTR)+
                80        CC    AD   9E 0004D          MOVAB   INPUT_BUFFER, ($$ITMBLKPTR)+
                                80   D4 00051          CLRL    ($$ITMBLKPTR)+
                80 000B01F4     8F   D0 00053          MOVL    #721396, ($$ITMBLKPTR)+
                80        24    AE   9E 0005A          MOVAB   BUFFER, ($$ITMBLKPTR)+
                                80   7C 0005E          CLRQ    ($$ITMBLKPTR)+
                                7E   7C 00060          CLRQ    -(SP)                           ; 0935
                          24    AE   9F 00062          PUSHAB  IOSB
                          0C    AE   9F 00065          PUSHAB  ITMLST
                7E                  07   7D 00068          MOVQ    #7, -(SP)
                                    7E   D4 0006B          CLRL    -(SP)
00000000G       00                  07   FB 0006D          CALLS   #7, SYS$SNDJBCW
                07                  50   E9 00074          BLBC    STATUS, 3$                      ; 0936
                50              1C  AE   D0 00077          MOVL    IOSB, STATUS                    ; 0937
                0D                  50   E8 0007B          BLBS    STATUS, 4$                      ; 0938
                                    53   DD 0007E 3$:      PUSHL   R3                              ; 0939
                                    7E   D4 00080          CLRL    -(SP)
                          0000'     CF   9F 00082          PUSHAB  P.AAE
0000V           CF                  03   FB 00086          CALLS   #3, STOP_BATCH_JOB
                52              24  AE   9E 0008B 4$:      MOVAB   BUFFER, PTR                     ; 0945
                                    62   D5 0008F 5$:      TSTL    (PTR)                           ; 0946
                                    1D   13 00091          BEQL    7$
                03              02  A2   B1 00093          CMPW    2(PTR), #3                      ; 0948
                                    0D   12 00097          BNEQ    6$
                                    53   DD 00099          PUSHL   R3                              ; 0949
                                28  AE   9F 0009B          PUSHAB  BUFFER
                                04  A2   9F 0009E          PUSHAB  4(PTR)
0000V           CF                  03   FB 000A1          CALLS   #3, STOP_BATCH_JOB
                50                  62   3C 000A6 6$:      MOVZWL  (PTR), R0                       ; 0950
                52              04 A042  9E 000A9          MOVAB   4(R0)[PTR], PTR
                                    DF   11 000AE          BRB     5$                              ; 0946
                                    04   00080 7$:         RET                                     ; 0954

; Routine Size:  177 bytes,    Routine Base:  $CODE$ + 0279
```

```
;  559          0955  1 ROUTINE stop_batch_job (flags, buffer, signal_args): NOVALUE =
;  560          0956  1
;  561          0957  1 !---
;  562          0958  1 !
;  563          0959  1 !       This routine is called to terminate a job stream as a result
;  564          0960  1 !       of an operator request or failure of an individual job step.
;  565          0961  1 !
;  566          0962  1 ! Inputs:
;  567          0963  1 !
;  568          0964  1 !       flags = Address of flags longword from job controller
;  569          0965  1 !       djt = Address of entire buffer from job controller
;  570          0966  1 !
;  571          0967  1 ! Outputs:
;  572          0968  1 !
;  573          0969  1 !       There is no return - the image is exited.
;  574          0970  1 !---
;  575          0971  1
;  576          0972  2 BEGIN
;  577          0973  2
;  578          0974  2 MAP
;  579          0975  2     flags : REF $BBLOCK,                    ! Address of options longword
;  580          0976  2     signal_args : REF $BBLOCK;              ! Address of signal arguments or 0
;  581          0977  2
;  582          0978  2 BIND
;  583          0979  2     lgi = .ppd [ppd$l_lgi]: BBLOCK;         ! Address the LGI area
;  584          0980  2
;  585          0981  2
;  586          0982  2 !
;  587          0983  2 ! Write the logout message.
;  588          0984  2 !
;  589          0985  2 logout_message();                           ! Write logout message
;  590          0986  2
;  591          0987  2 !
;  592          0988  2 !
;  593          0989  2 IF .flags[dji$v_log_delete]                 ! If delete of output file requested
;  594          0990  2 AND NOT .flags[dji$v_log_spool]             ! and no print,
;  595          0991  2 THEN
;  596          0992  2     output_fab [fab$v_dlt] = true;          ! then set to delete on close
;  597          0993  2
;  598          0994  2 $CMEXEC(ROUTIN = close_output);             ! Close log file so we can print it
;  599          0995  2
;  600          0996  2 $CMKRNL(ROUTIN = set_uic, ARGLST = .lgi [lgi$l_origuic]); ! Reset UIC
;  601          0997  2
;  602          0998  2 IF .flags[dji$v_lcg_spool]                  ! If output file is to be printed
;  603          0999  2 THEN
;  604          1000  3     BEGIN
;  605          1001  3     LOCAL
;  606          1002  3         wrdptr : REF VECTOR[,WORD],
;  607          1003  3         ptr : REF VECTOR,                   ! Pointer to item list
;  608          1004  3         queue_name : VECTOR[2]              ! Place for queue name
;  609          1005  3                      INITIAL(0,0),
;  610          1006  3         job_name : VECTOR[2]                ! Place for job name
;  611          1007  3                    INITIAL(0,0),
;  612          1008  3         itmlst : VECTOR[30],                ! Item list for SNDJBC
;  613          1009  3         input_buffer : VECTOR[128];         ! Batch input item
;  614          1010  3
;  615          1011  3 !
```

```
;  616     1012  3 ! We need to find the queue name, as well as the job name, before starting
;  617     1013  3 ! to fill out the itemlist.
;  618     1014  3 !
;  619     1015  3     wrdptr = .buffer;                   ! Point at the buffer
;  620     1016  3     WHILE true DO                       ! Go thru buffer
;  621     1017  4         BEGIN
;  622     1018  4         IF .wrdptr[1] EQL 0
;  623     1019  4         THEN EXITLOOP;
;  624     1020  4         IF .wrdptr[1] EQL dji$k_log_queue
;  625     1021  4         THEN
;  626     1022  5             BEGIN
;  627     1023  5             queue_name[0] = .wrdptr[0];
;  628     1024  5             queue_name[1] = wrdptr[2];
;  629     1025  5             IF .job_name[1] NEQ 0
;  630     1026  5             THEN EXITLOOP;
;  631     1027  5             END
;  632     1028  4         ELSE IF .wrdptr[1] EQL dji$k_job_name
;  633     1029  4         THEN
;  634     1030  5             BEGIN
;  635     1031  5             job_name[0] = .wrdptr[0];
;  636     1032  5             job_name[1] = wrdptr[2];
;  637     1033  5             IF .queue_name[1] NEQ 0
;  638     1034  5             THEN EXITLOOP;
;  639     1035  4             END;
;  640     1036  4         wrdptr = wrdptr[2] + .wrdptr[0];
;  641     1037  3         END;
;  642     1038  3 !
;  643     1039  3 ! If no queue name was found, then use SYS$PRINT.
;  644     1040  3 !
;  645     1041  3     IF .queue_name[0] EQL 0
;  646     1042  3     THEN
;  647     1043  4         BEGIN
;  648     1044  4         queue_name[0] = %CHARCOUNT('SYS$PRINT');
;  649     1045  4         queue_name[1] = UPLIT BYTE('SYS$PRINT');
;  650     1046  3         END;
;  651     1047  3 !
;  652     1048  3 ! Now to fill in the itemlist.
;  653     1049  3 !
;  654     1050  3     ptr = itmlst;                       ! Start at beginning of item list
;  655     1051  3 !
;  656     1052  3 ! The queue name is either in the JBC buffer, or else we should use
;  657     1053  3 ! SYS$PRINT.
;  658     1054  3 !
;  659     1055  3     ptr[0] = sjc$_queue^16 OR .queue_name[0];
;  660     1056  3     ptr[1] = .queue_name[1];
;  661     1057  3     ptr[2] = 0;
;  662     1058  3     ptr = ptr[3];
;  663     1059  3 !
;  664     1060  3 ! Also put in the job name.
;  665     1061  3 !
;  666     1062  3     ptr[0] = sjc$_job_name^16 OR .job_name[0];
;  667     1063  3     ptr[1] = .job_name[1];
;  668     1064  3     ptr[2] = 0;
;  669     1065  3     ptr = ptr[3];
;  670     1066  3 !
;  671     1067  3 ! Add /NOTIFY if requested.
;  672     1068  3 !
```

```
;  673          1069  3     IF .flags[dji$v_notify]
;  674          1070  3     THEN
;  675          1071  4         BEGIN
;  676          1072  4         ptr[0] = sjc$_notify^16;
;  677          1073  4         ptr[1] = ptr[2] = 0;
;  678          1074  4         ptr = ptr[3];
;  679          1075  3         END;
;  680          1076  3 !
;  681          1077  3 ! If the log file exists, add its information.
;  682          1078  3 !
;  683          1079  3     IF CH$RCHAR(ppd[ppd$t_outdvi]) NEQ 0
;  684          1080  3     THEN
;  685          1081  4         BEGIN
;  686          1082  4 !
;  687          1083  4 ! File ID
;  688          1084  4 !
;  689          1085  4         ptr[0] = sjc$_file_identification^16 OR ppd$c_dvifid;
;  690          1086  4         ptr[1] = ppd[ppd$t_outdvi];
;  691          1087  4         ptr[2] = 0;
;  692          1088  4         ptr = ptr[3];
;  693          1089  4 !
;  694          1090  4 ! Add /DELETE if requested.
;  695          1091  4 !
;  696          1092  4         IF .flags[dji$v_log_delete]
;  697          1093  4         THEN
;  698          1094  5             BEGIN
;  699          1095  5             ptr[0] = sjc$_delete_file^16;
;  700          1096  5             ptr[1] = ptr[2] = 0;
;  701          1097  5             ptr = ptr[3];
;  702          1098  4             END;
;  703          1099  4 !
;  704          1100  4 ! The log file always gets a header page.
;  705          1101  4 !
;  706          1102  4         ptr[0] = sjc$_page_header^16;
;  707          1103  4         ptr[1] = ptr[2] = 0;
;  708          1104  4         ptr = ptr[3];
;  709          1105  4         END
;  710          1106  4 !
;  711          1107  4 ! If no log file exists, attempt to print messages.
;  712          1108  4 !
;  713          1109  3     ELSE IF .signal_args NEQA 0
;  714          1110  3     THEN
;  715          1111  4         BEGIN
;  716          1112  4         LOCAL
;  717          1113  4             i : REF $BBLOCK,     ! Pointer to item header
;  718          1114  4             p : REF $BBLOCK;     ! Pointer to input buffer
;  719          1115  4
;  720          1116  4         p = i = input_buffer;
;  721          1117  4         p[dji$w_item_code] = dji$k_condition_vector;
;  722          1118  4         p[dji$w_item_size] = 4;
;  723          1119  4         p = .p + dji$s_item_header;
;  724          1120  4         .p = .signal_args[4,0,32,0];            ! Primary condition
;  725          1121  4         p = .p + 4;
;  726          1122  4         IF .signal_args[0,0,8,0] GEQU 3
;  727          1123  4         THEN
;  728          1124  5             BEGIN
;  729          1125  5             i[dji$w_item_size] = 8;
```

```
;  730     1126  5                .p = .signal_args[12,0,32,0];       ! STS, if present
;  731     1127  5                p = .p + 4;
;  732     1128  4                END;
;  733     1129  4            IF .signal_args[0,0,8,0] GEQU 4
;  734     1130  4            THEN
;  735     1131  5                BEGIN
;  736     1132  5                i[dji$w_item_size] = 12;
;  737     1133  5                .p = .signal_args[16,0,32,0];       ! STV, if present
;  738     1134  5                p = .p + 4;
;  739     1135  4                END;
;  740     1136  4
;  741     1137  4            IF .output_nam[nam$b_rsl] NEQ 0
;  742     1138  4            THEN
;  743     1139  5                BEGIN
;  744     1140  5                p[dji$w_item_code] = dji$k_file_specification;
;  745     1141  5                p[dji$w_item_size] = .output_nam[nam$b_rsl];
;  746     1142  5                p = .p + dji$s_item_header;
;  747     1143  5                p = CH$MOVE(
;  748     1144  5                    .output_nam[nam$b_rsl],
;  749     1145  5                    .output_nam[nam$l_rsa],
;  750     1146  5                    .p);
;  751     1147  5                END
;  752     1148  4            ELSE IF .output_nam[nam$b_esl] NEQ 0
;  753     1149  4            THEN
;  754     1150  5                BEGIN
;  755     1151  5                p[dji$w_item_code] = dji$k_file_specification;
;  756     1152  5                p[dji$w_item_size] = .output_nam[nam$b_esl];
;  757     1153  5                p = .p + dji$s_item_header;
;  758     1154  5                p = CH$MOVE(
;  759     1155  5                    .output_nam[nam$b_esl],
;  760     1156  5                    .output_nam[nam$l_esa],
;  761     1157  5                    .p);
;  762     1158  5                END
;  763     1159  4            ELSE
;  764     1160  5                BEGIN
;  765     1161  5                p[dji$w_item_code] = dji$k_file_specification;
;  766     1162  5                p[dji$w_item_size] = .output_fab[fab$b_fns];
;  767     1163  5                p = .p + dji$s_item_header;
;  768     1164  5                p = CH$MOVE(
;  769     1165  5                    .output_fab[fab$b_fns],
;  770     1166  5                    .output_fab[fab$l_fna],
;  771     1167  5                    .p);
;  772     1168  4                END;
;  773     1169  4
;  774     1170  4            .p = 0;                                     ! Zero terminate list
;  775     1171  4            p = .p + 4;
;  776     1172  4
;  777     1173  4            ptr[0] = sjc$_batch_input^16 OR (.p - input_buffer);
;  778     1174  4            ptr[1] = input_buffer;
;  779     1175  4            ptr[2] = 0;
;  780     1176  4            ptr = ptr[3];
;  781     1177  3            END;
;  782     1178  3 !
;  783     1179  3 ! Done.  Put in a zero longowrd
;  784     1180  3 !
;  785     1181  3     ptr[0] = 0;
;  786     1182  3
```

```
;  787        P 1183  3     $SNDJBCW(FUNC = sjc$_enter_file,
;  788          1184  3              ITMLST = itmlst);
;  789          1185  2     END;
;  790          1186  2
;  791          1187  2 $CMEXEC(ROUTIN = exit_process);          ! Terminate process
;  792          1188  2
;  793          1189  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                  54  4E  49  52  50  24  53  59  53  00024 P.AAF:  .ASCII  \SYS$PRINT\                                ;

                                                        .EXTRN  SYS$CMEXEC, SYS$CMKRNL

                                                        .PSECT  $CODE$,NOWRT,2

                                                07FC 00000 STOP_BATCH_JOB:
                                                        .WORD   Save ,r2,R3,R4,R5,R6,R7,R8,R9,R10               ; 0955
                                    5A 00000000G   00   9E 00002        MOVAB   SYS$CMEXEC, R10                 ;
                                    59 00000000G   00   9E 00009        MOVAB   PPD+72, R9                      ;
                                    58 00000000G   00   9E 00010        MOVAB   OUTPUT_FAB+4, R8                ;
                                    57 00000000G   00   9E 00017        MOVAB   OUTPUT_NAM+3, R7                ;
                                    5E      FD78   CE   9E 0001E        MOVAB   -648(SP), SP                    ;
                                    52        CC   A9   D0 00023        MOVL    PPD+20, R2                      ; 0979
                     00000000G      00           00   FB 00027        CALLS   #0, LOGOUT_MESSAGE              ; 0985
                  0A        04      BC           01   E1 0002E        BBC     #1, @FLAGS, 1$                  ; 0989
                  05        04      BC           03   E0 00033        BBS     #3, @FLAGS, 1$                  ; 0990
                            01      A8         80 8F   88 00038        BISB2   #128, OUTPUT_FAB+5              ; 0992
                                                7E   D4 0003D 1$:     CLRL    -(SP)                           ; 0994
                                 00000000G      00   9F 0003F        PUSHAB  CLOSE_OUTPUT                    ;
                                    6A          02   FB 00045        CALLS   #2, SYS$CMEXEC                  ;
                                                62   DD 00048        PUSHL   (R2)                            ; 0996
                                 00000000G      00   9F 0004A        PUSHAB  SET_UIC                         ;
                     00000000G      00           02   FB 00050        CALLS   #2, SYS$CMKRNL                  ;
               03        04      BC           03   E0 00057        BBS     #3, @FLAGS, 2$                  ; 0998
                                           0154 31   0005C        BRW     20$                             ;
                                       F8  AD   7C   0005F 2$:     CLRQ    QUEUE_NAME                      ; 1000
                                       F0  AD   7C   00062        CLRQ    JOB_NAME                        ;
                                    50     08   AC   D0 00065        MOVL    BUFFER, WRDPTR                  ; 1015
                                    51     02   A0   3C 00069 3$:     MOVZWL  2(WRDPTR), R1                   ; 1018
                                                30   13 0006D        BEQL    7$                              ;
                                    05          51   B1 0006F        CMPW    R1, #5                          ; 1020
                                                0E   12 00072        BNEQ    4$                              ;
                                 F8 AD          60   3C 00074        MOVZWL  (WRDPTR), QUEUE_NAME            ; 1023
                                 FC AD       04 A0   9E 00078        MOVAB   4(R0), QUEUE_NAME+4             ; 1024
                                             F4 AD   D5 0007D        TSTL    JOB_NAME+4                      ; 1025
                                                11   11 00080        BRB     5$                              ;
                                    04          51   B1 00082 4$:     CMPW    R1, #4                          ; 1028
                                                0E   12 00085        BNEQ    6$                              ;
                                 F0 AD          60   3C 00087        MOVZWL  (WRDPTR), JOB_NAME              ; 1031
                                 F4 AD       04 A0   9E 0008B        MOVAB   4(R0), JOB_NAME+4               ; 1032
                                             FC AD   D5 00090        TSTL    QUEUE_NAME+4                    ; 1033
                                                0A   12 00093 5$:     BNEQ    7$                              ;
                                    51          60   3C 00095 6$:     MOVZWL  (WRDPTR), R1                    ; 1036
                                    50     04 A140   9E 00098        MOVAB   4(R1)[WRDPTR], WRDPTR           ;
```

```
                                    CA  11 0009D          BRB     3$                                      ; 1016
                                F8  AD  D5 0009F 7$:      TSTL    QUEUE_NAME                              ; 1041
                                    0A  12 000A2          BNEQ    8$                                      ;
                            F8  AD      09 D0 000A4       MOVL    #9, QUEUE_NAME                          ; 1044
                            FC  AD  0000' CF 9E 000A8     MOVAB   P.AAF, QUEUE_NAME+4                     ; 1045
                            56      FF78  CD 9E 000AE 8$: MOVAB   ITMLST, PTR                            ; 1050
                    86      F8  AD 00860000 8F C9 000B3   BISL3   #8781824, QUEUE_NAME, (PTR)+            ; 1055
                                86      FC  AD D0 000BC   MOVL    QUEUE_NAME+4, (PTR)+                    ; 1056
                                            86 D4 000C0   CLRL    (PTR)+                                  ; 1057
                    86      F0  AD 004F0000 8F C9 000C2   BISL3   #5177344, JOB_NAME, (PTR)+              ; 1062
                                86      F4  AD D0 000CB   MOVL    JOB_NAME+4, (PTR)+                      ; 1063
                                            86 D4 000CF   CLRL    (PTR)+                                  ; 1064
                09      04  BC              04 E1 000D1   BBC     #4, @FLAGS, 9$                          ; 1069
                                86 006C0000 8F D0 000D6   MOVL    #7077888, (PTR)+                        ; 1072
                                            86 7C 000DD   CLRQ    (PTR)+                                  ; 1073
                                            69 95 000DF 9$: TSTB  PPD+72                                  ; 1079
                                            27 13 000E1   BEQL    11$                                     ;
                                86 0027001C 8F D0 000E3   MOVL    #2555932, (PTR)+                        ; 1085
                                86          69 9E 000EA   MOVAB   PPD+72, (PTR)+                          ; 1086
                                            86 D4 000ED   CLRL    (PTR)+                                  ; 1087
                09      04  BC              01 E1 000EF   BBC     #1, @FLAGS, 10$                         ; 1092
                                86 00180000 8F D0 000F4   MOVL    #1572864, (PTR)+                        ; 1095
                                            86 7C 000FB   CLRQ    (PTR)+                                  ; 1096
                                66 00710000 8F D0 000FD 10$:  MOVL    #7405568, (PTR)                         ; 1102
                                        04  A6 7C 00104   CLRQ    4(PTR)                                  ; 1103
                                    0090    31 00107      BRW     18$                                     ; 1104
                                50      0C  AC D0 0010A 11$:  MOVL    SIGNAL_ARGS, R0                         ; 1109
                                            03 12 0010E   BNEQ    12$                                     ;
                                        008A 31 00110     BRW     19$                                     ;
                                51          6E 9E 00113 12$:  MOVAB   INPUT_BUFFER, I                     ; 1116
                                53          51 D0 00116   MOVL    I, P                                    ;
                                83 80020004 8F D0 00119   MOVL    #-2147352572, (P)+                      ; 1118
                                83      04  A0 D0 00120   MOVL    4(R0), (P)+                             ; 1120
                                03          60 91 00124   CMPB    (R0), #3                                ; 1122
                                            07 1F 00127   BLSSU   13$                                     ;
                                61          08 B0 00129   MOVW    #8, (I)                                 ; 1125
                                83      0C  A0 D0 0012C   MOVL    12(R0), (P)+                            ; 1126
                                04          60 91 00130 13$:  CMPB    (R0), #4                                ; 1129
                                            07 1F 00133   BLSSU   14$                                     ;
                                61          0C B0 00135   MOVW    #12, (I)                                ; 1132
                                83      10  A0 D0 00138   MOVL    16(R0), (P)+                            ; 1133
                                50          67 9A 0013C 14$:  MOVZBL  OUTPUT_NAM+3, R0                    ; 1137
                                            12 13 0013F   BEQL    15$                                     ;
                        02  A3      8003    8F B0 00141   MOVW    #-32765, 2(P)                           ; 1140
                                83          50 B0 00147   MOVW    R0, (P)+                                ; 1141
                                53          02 C0 0014A   ADDL2   #2, P                                   ; 1142
                                51      01  A7 D0 0014D   MOVL    OUTPUT_NAM+4, R1                        ; 1145
                                            2C 11 00151   BRB     17$                                     ; 1146
                                50      08  A7 9A 00153 15$:  MOVZBL  OUTPUT_NAM+11, R0                  ; 1148
                                            12 13 00157   BEQL    16$                                     ;
                        02  A3      8003    8F B0 00159   MOVW    #-32765, 2(P)                           ; 1151
                                83          50 B0 0015F   MOVW    R0, (P)+                                ; 1152
                                53          02 C0 00162   ADDL2   #2, P                                   ; 1153
                                51      09  A7 D0 00165   MOVL    OUTPUT_NAM+12, R1                       ; 1156
                                            14 11 00169   BRB     17$                                     ; 1157
                        02  A3      8003    8F B0 0016B 16$:  MOVW    #-32765, 2(P)                           ; 1161
                                50      30  A8 9A 00171   MOVZBL  OUTPUT_FAB+52, R0                       ; 1162
```

```
                                83              50  B0 00175          MOVW    R0, (P)+                                    ;
                                53              02  C0 00178          ADDL2   #2, P                                       ; 1163
                                51        28    A8  D0 0017B          MOVL    OUTPUT_FAB+44, R1                           ; 1166
                         63     61              50  28 0017F 17$:     MOVC3   R0, (R1), (P)                               ; 1167
                                83              D4  00183             CLRL    (P)+                                        ; 1170
                                50              6E  9E 00185          MOVAB   INPUT_BUFFER, R0                            ; 1173
                                53              50  C2 00188          SUBL2   R0, R3
                         66     53  000A0000    8F  C9 0018B          BISL3   #655360, R3, (PTR)
                            04  A6              6E  9E 00193          MOVAB   INPUT_BUFFER, 4(PTR)                        ; 1174
                                        08      A6  D4 00197          CLRL    8(PTR)                                      ; 1175
                                56              0C  C0 0019A 18$:     ADDL2   #12, PTR                                    ; 1176
                                                66  D4 0019D 19$:     CLRL    (PTR)                                       ; 1181
                                                7E  7C 0019F          CLRQ    -(SP)                                       ; 1184
                                                7E  D4 001A1          CLRL    -(SP)                                       ;
                                        FF78    CD  9F 001A3          PUSHAB  ITMLST                                      ;
                                7E              13  7D 001A7          MOVQ    #19, -(SP)                                  ;
                                                7E  D4 001AA          CLRL    -(SP)                                       ;
                   00000000G    00              07  FB 001AC          CALLS   #7, SYS$SNDJBCW                             ;
                                                7E  D4 001B3 20$:     CLRL    -(SP)                                       ; 1187
                                    00000000G   00  9F 001B5          PUSHAB  EXIT_PROCESS                                ;
                                6A              02  FB 001BB          CALLS   #2, SYS$CMEXEC                              ;
                                                    04 001BE          RET                                                 ; 1189

; Routine Size:  447 bytes,    Routine Base:  $CODE$ + 032A
```

```
;  795          1190  1 GLOBAL ROUTINE init_network: NOVALUE =
;  796          1191  1
;  797          1192  1 !---
;  798          1193  1 !
;  799          1194  1 !       Perform initializations of a network job.  The primary logical
;  800          1195  1 !       names have been setup by the calling process (NETACP).
;  801          1196  1 !
;  802          1197  1 ! Inputs:
;  803          1198  1 !
;  804          1199  1 !       sys$input = Name of command procedure to be invoked
;  805          1200  1 !       sys$output = 3 concatenated ASCIC access control strings
;  806          1201  1 !                    (username, password, account)
;  807          1202  1 !       sys$error = Network control block (NCB) for connect
;  808          1203  1 !
;  809          1204  1 ! Outputs:
;  810          1205  1 !
;  811          1206  1 !       uaf_record = Address of UAF record, if any
;  812          1207  1 !---
;  813          1208  1
;  814          1209  2 BEGIN
;  815          1210  2
;  816          1211  2 LOCAL
;  817          1212  2     len,
;  818          1213  2     ptr,
;  819          1214  2     username:   VECTOR [2],          ! Username descriptor
;  820          1215  2     password:   VECTOR [2],          ! Password descriptor
;  821          1216  2     account:    VECTOR [2];          ! Account descriptor
;  822          1217  2
;  823          1218  2 job_type = jib$c_network;           ! Say this is a network job
;  824          1219  2
;  825          1220  2 !
;  826          1221  2 ! Get logical link number from NCB and store remote node info in P1 space
;  827          1222  2 !
;  828          1223  2
;  829          1224  2 IF NOT CH$FAIL(ptr = CH$FIND_CH(.sys$error [0], .sys$error [1], '/'))
;  830          1225  2 THEN
;  831          1226  2     set_node_name(.(.ptr + 1)<0,16>);   ! Set remote node info in P1 space
;  832          1227  2
;  833          1228  2 !
;  834          1229  2 ! Check to see if this network job should use the proxy login.  This is
;  835          1230  2 ! determined by checking the low bit of the first word of the string
;  836          1231  2 ! described by the sys$output descriptor.  If the bit is set, then attempt
;  837          1232  2 ! a proxy login.  If that fails, tnen use the access control string described
;  838          1233  2 ! by the sys$output descriptor.
;  839          1234  2 !
;  840          1235  2
;  841          1236  2 ptr = ..sys$output[1];                ! Get first two words of SYS$OUTPUT string
;  842          1237  2
;  843          1238  2 sys$output[0] = .sys$output[0] - 2;    ! Drop flag word off SYS$OUTPUT string
;  844          1239  2 sys$output[1] = .sys$output[1] + 2;
;  845          1240  2
;  846          1241  2 IF NOT
;  847          1242  3     BEGIN
;  848          1243  3     IF .ptr<0,1> THEN get_proxy()   ! If low bit set then try proxy login
;  849          1244  3     ELSE FALSE                      ! Otherwise set FALSE
;  850          1245  3     END
;  851          1246  2 THEN                                ! If no proxy, or proxy failed
```

```
;  852     1247  3     BEGIN                                   ! Use the access control string
;  853     1248  3     ptr = .sys$output [1];                  ! Get address of SYS$OUTPUT string
;  854     1249  3
;  855     1250  3     username [0] = CH$RCHAR_A(ptr);         ! Get length of username
;  856     1251  3     username [1] = .ptr;                    !  and address of username
;  857     1252  3
;  858     1253  3     ptr = .ptr + .username [0];             ! Skip to password
;  859     1254  3     password [0] = CH$RCHAR_A(ptr);         ! Get length of password
;  860     1255  3     password [1] = .ptr;                    !  and address of password
;  861     1256  3
;  862     1257  3     ptr = .ptr + .password [0];             ! Skip to account
;  863     1258  3     account [0] = CH$RCHAR_A(ptr);          ! Get length of account
;  864     1259  3     account [1] = .ptr;                     !  and address of account
;  865     1260  3
;  866     1261  3     IF NOT .pcb_sts [$BITPOSITION(pcb$v_login)]
;  867     1262  3     THEN validate_uafrec(username,          ! Lookup in UAF file
;  868     1263  3                          password,          !  and validate the password
;  869     1264  3                          UPLIT (0,0));      !   with a null secondary password
;  870     1265  2     END;
;  871     1266  2
;  872     1267  2 !
;  873     1268  2 ! Create SYS$NET logical name with contents of NCB
;  874     1269  2 !
;  875     1270  2
;  876     1271  2 create_logical(%ASCID 'SYS$NET',         ! Define SYS$NET
;  877     1272  2                sys$error,
;  878     1273  2                psl$c_exec);
;  879     1274  2
;  880     1275  2 !
;  881     1276  2 ! If the input file has the file type .EXE, then rather than activating
;  882     1277  2 ! the CLI and creating a log file, activate the program from a small
;  883     1278  2 ! code segment in P1 space.  This is done to optimize network job
;  884     1279  2 ! activation time.
;  885     1280  2 !
;  886     1281  2
;  887     1282  2 IF NOT CH$FAIL(CH$FIND_SUB(.sys$input [0], .sys$input [1],
;  888     1283  2                             4, UPLIT BYTE('.EXE')))
;  889     1284  2 THEN
;  890     1285  3     BEGIN
;  891     1286  3     $CMEXEC(ROUTIN = map_imgact);          ! Map the imgact code segment into P1
;  892     1287  3     input_fab [fab$l_fna] = UPLIT BYTE('_NL:');          ! Set input to NL:
;  893     1288  3     input_fab [fab$b_fns] = 4;
;  894     1289  3     output_fab [fab$l_fna] = .input_fab [fab$l_fna];     ! Set output to NL:
;  895     1290  3     output_fab [fab$b_fns] = .input_fab [fab$b_fns];
;  896     1291  3     RETURN;                                ! and return
;  897     1292  2     END;
;  898     1293  2
;  899     1294  2 !
;  900     1295  2 ! Set default filespec for input file
;  901     1296  2 !
;  902     1297  2
;  903     1298  2 input_fab [fab$l_dna] = UPLIT BYTE('CONNECT.COM');
;  904     1299  2 input_fab [fab$b_dns] = 11;
;  905     1300  2
;  906     1301  2 !
;  907     1302  2 ! Construct filespec of output log file for network job
;  908     1303  2 !
```

```
;  909          1304  2
;  910          1305  2 ptr = (sys$output [1] = .sys$input [1]);
;  911          1306  2 IF (len = (sys$output [0] = .sys$input [0])) NEQ 0
;  912          1307  2 THEN
;  913          1308  3     BEGIN
;  914          1309  3     DO
;  915          1310  4         BEGIN
;  916          1311  4         LOCAL
;  917          1312  4             chr: BYTE;
;  918          1313  4         chr = CH$RCHAR_A(ptr);
;  919          1314  4         IF .chr EQL ':'
;  920          1315  4         OR .chr EQL ']'
;  921          1316  4         OR .chr EQL '>'
;  922          1317  4         THEN
;  923          1318  5             BEGIN
;  924          1319  5             sys$output [1] = .ptr;
;  925          1320  5             sys$output [0] = .len - 1;
;  926          1321  4             END;
;  927          1322  4         len = .len - 1;
;  928          1323  4         END
;  929          1324  3     WHILE .len GTR 0;
;  930          1325  3     IF NOT CH$FAIL(ptr = CH$FIND_CH(.sys$output [0], .sys$output [1], '.'))
;  931          1326  3     THEN
;  932          1327  3         sys$output [0] = CH$DIFF(.ptr, .sys$output [1]);
;  933          1328  2     END;
;  934          1329  2
;  935          1330  2 output_fab [fab$b_fns] = .sys$output [0]; ! Set as primary output filespec
;  936          1331  2 output_fab [fab$l_fna] = .sys$output [1];
;  937          1332  2
;  938          1333  1 END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                00002D          .BLKB   3
                                    00000000  00000000  00030 P.AAG:  .LONG   0, 0                                          ;
                                00  54  45  4E  24  53  59  53  00038 P.AAI:  .ASCII  \SYS$NET\<0>                           ;
                                                010E0007  00040 P.AAH:  .LONG   17694727                                    ;
                                                00000000'  00044        .ADDRESS P.AAI                                      ;
                                            45  58  45  2E  00048 P.AAJ:  .ASCII  \.EXE\                                    ;
                                            3A  4C  4E  5F  0004C P.AAK:  .ASCII  \_NL:\                                    ;
4D  4F  43  2E  54  43  45  4E  4E  4F  43  00050 P.AAL:  .ASCII  \CONNECT.COM\                                              ;


                                                    .PSECT  $CODE$,NOWRT,2

                                            07FC 00000      .ENTRY  INIT_NETWORK, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1190
                                                            R10                                                 ;
                                5A      0000'   CF  9E 00002    MOVAB   P.AAG, R10                              ;
                                59 00000000G    00  9E 00007    MOVAB   SYS$ERROR, R9                           ;
                                58 00000000G    00  9E 0000E    MOVAB   OUTPUT_FAB+44, R8                       ;
                                57 00000000G    00  9E 00015    MOVAB   SYS$INPUT+4, R7                         ;
                                56 00000000G    00  9E 0001C    MOVAB   INPUT_FAB+44, R6                        ;
                                55 00000000G    00  9E 00023    MOVAB   SYS$OUTPUT+4, R5                        ;
                                5E              18  C2 0002A    SUBL2   #24, SP                                 ;
```

```
                00000000G  00          01  DO 0002D          MOVL    #1, JOB_TYPE                        ; 1218
                           50       0-  A9  DO 00034          MOVL    SYS$ERROR+4, R0                     ; 1224
                60         69          2F  3A 00038          LOCC    #47, SYS$ERROR, (R0)                ;
                                       02  12 0003C          BNEQ    1$                                  ;
                                       51  D4 0003E          CLRL    R1                                  ;
                           54          51  DO 00040 1$:      MOVL    R1, PTR                             ;
                                       0B  13 00043          BEQL    2$                                  ;
                           7E       01 A4  3C 00045          MOVZWL  1(PTR), -(SP)                       ; 1226
                00000000G  00          01  FB 00049          CALLS   #1, SET_NODE_NAME                   ;
                           50          65  DO 00050 2$:      MOVL    SYS$OUTPUT+4, R0                    ; 1236
                           54          60  DO 00053          MOVL    (R0), PTR                           ;
                              FC       A5  02  C2 00056          SUBL2   #2, SYS$OUTPUT                      ; 1238
                           65          02  C0 0005A          ADDL2   #2, SYS$OUTPUT+4                    ; 1239
                           08          54  E9 0005D          BLBC    PTR, 3$                             ; 1243
                     0000V CF          00  FB 00060          CALLS   #0, GET_PROXY                       ;
                           39          50  E8 00065          BLBS    R0, 4$                              ;
                           54          65  DO 00068 3$:      MOVL    SYS$OUTPUT+4, PTR                   ; 1248
                              10       AE  84  9A 0006B          MOVZBL  (PTR)+, USERNAME                    ; 1250
                              14       AE  54  DO 0006F          MOVL    PTR, USERNAME+4                     ; 1251
                           54          10  AE  C0 00073          ADDL2   USERNAME, PTR                       ; 1253
                              08       AE  84  9A 00077          MOVZBL  (PTR)+, PASSWORD                    ; 1254
                              0C       AE  54  DO 0007B          MOVL    PTR, PASSWORD+4                     ; 1255
                           54          08  AE  C0 0007F          ADDL2   PASSWORD, PTR                       ; 1257
                           6E          84  9A 00083          MOVZBL  (PTR)+, ACCOUNT                     ; 1258
                              04       AE  54  DO 00086          MOVL    PTR, ACCOUNT+4                      ; 1259
                0F 00000000G  00          04  E0 0008A          BBS     #4, PCB_STS+2, 4$                   ; 1261
                                          5A  DD 00092          PUSHL   R10                                 ; 1264
                                       0C  AE  9F 00094          PUSHAB  PASSWORD                            ; 1262
                                       18  AE  9F 00097          PUSHAB  USERNAME                            ;
                00000000G  00              03  FB 0009A          CALLS   #3, VALIDATE_UAFREC                 ;
                                           01  DD 000A1 4$:      PUSHL   #1                                  ; 1271
                                           59  DD 000A3          PUSHL   R9                                  ;
                                       10  AA  9F 000A5          PUSHAB  P.AAH                               ;
                00000000G  00              03  FB 000A8          CALLS   #3, CREATE_LOGICAL                  ;
                           50              67  DO 000AF          MOVL    SYS$INPUT+4, R0                     ; 1282
                60      FC    A7     18  AA              04  39 000B2          MATCHC  #4, P.AAJ, SYS$INPUT, (R0)  ; 1283
                                           03  13 000B9          BEQL    5$                                  ;
                           53              04  DO 000BB          MOVL    #4, R3                              ;
                           53              04  C2 000BE 5$:      SUBL2   #4, R3                              ;
                                           20  13 000C1          BEQL    6$                                  ;
                                           7E  D4 000C3          CLRL    -(SP)                               ; 1286
                                  00000000G    00  9F 000C5          PUSHAB  MAP_IMGACT                          ;
                00000000G  00              02  FB 000CB          CALLS   #2, SYS$CMEXEC                      ;
                                       66      1C  AA  9E 000D2          MOVAB   P.AAK, INPUT_FAB+44         ; 1287
                                 08    A6          04  90 000D6          MOVB    #4, INPUT_FAB+52            ; 1288
                                       68          66  DO 000DA          MOVL    INPUT_FAB+44, OUTPUT_FAB+44 ; 1289
                                 08    A8      08  A6  90 000DD          MOVB    INPUT_FAB+52, OUTPUT_FAB+52 ; 1290
                                                   04  000E2          RET                                 ; 1285
                                 04    A6      20  AA  9E 000E3 6$:      MOVAB   P.AAL, INPUT_FAB+48         ; 1298
                                 09    A6          0B  90 000E8          MOVB    #11, INPUT_FAB+53           ; 1299
                                       50          67  DO 000EC          MOVL    SYS$INPUT+4, R0             ; 1305
                                       65          50  DO 000EF          MOVL    R0, SYS$OUTPUT+4            ;
                                       54          50  DO 000F2          MOVL    R0, PTR                     ;
                                       50      FC  A7  DO 000F5          MOVL    SYS$INPUT, R0               ; 1306
                                 FC    A5          50  DO 000F9          MOVL    R0, SYS$OUTPUT              ;
                                                   34  13 000FD          BEQL    11$                         ;
                                       51          84  90 000FF 7$:      MOVB    (PTR)+, CHR                 ; 1313
```

```
                        3A          51  91 00102          CMPB    CHR, #58                                ; 1314
                                    0B  13 00105          BEQL    8$
                    5D  8F          51  91 00107          CMPB    CHR, #93                                ; 1315
                                    05  13 0010B          BEQL    8$
                        3E          51  91 0010D          CMPB    CHR, #62                                ; 1316
                                    08  12 00110          BNEQ    9$
                        65          54  D0 00112 8$:      MOVL    PTR, SYS$OUTPUT+4                       ; 1319
                    FC  A5      FF  A0  9E 00115          MOVAB   -1(R0), SYS$OUTPUT                      ; 1320
                        E2          50  F5 0011A 9$:      SOBGTR  LEN, 7$                                 ; 1322
                        52          65  D0 0011D          MOVL    SYS$OUTPUT+4, R2                        ; 1325
                62      FC  A5      2E  3A 00120          LOCC    #46, SYS$OUTPUT, (R2)
                                    02  12 00125          BNEQ    10$
                                    51  D4 00127          CLRL    R1
                        54          51  D0 00129 10$:     MOVL    R1, PTR
                                    05  13 0012C          BEQL    11$
            FC  A5                  52  C3 0012E          SUBL3   R2, PTR, SYS$OUTPUT                     ; 1327
                    08  A8      FC  A5  90 00133 11$:     MOVB    SYS$OUTPUT, OUTPUT_FAB+52               ; 1330
                        68          65  D0 00138          MOVL    SYS$OUTPUT+4, OUTPUT_FAB+44             ; 1331
                                    04  00013B           RET                                             ; 1333

; Routine Size:  316 bytes,    Routine Base:  $CODE$ + 04E9
```

```
;  940          1334  1 ROUTINE get_proxy =
;  941          1335  1 !---
;  942          1336  1 !
;  943          1337  1 !       Get the local username that is mapped to the remote username.
;  944          1338  1 !       The remote username is contained in the NCB string described
;  945          1339  1 !       by NCB_DESC, the NCB descriptor.
;  946          1340  1 !
;  947          1341  1 ! Inputs:
;  948          1342  1 !
;  949          1343  1 !       sys$error = address of NCB descriptor
;  950          1344  1 !
;  951          1345  1 ! Outputs:
;  952          1346  1 !
;  953          1347  1 !       uaf_record = Address of UAF record, if any
;  954          1348  1 !
;  955          1349  1 ! Status returns:
;  956          1350  1 !
;  957          1351  1 !       TRUE => Proxy username found
;  958          1352  1 !       FALSE => No proxy username found
;  959          1353  1 !
;  960          1354  1 !---
;  961          1355  1
;  962          1356  2 BEGIN
;  963          1357  2
;  964          1358  2 LOCAL
;  965          1359  2     status,
;  966          1360  2     netfab : BBLOCK[fab$c_bln],                 ! Fab for NETUAF.DAT
;  967          1361  2     netrab : BBLOCK[rab$c_bln],                 ! Rab for NETUAF.DAT
;  968          1362  2     net_record : BBLOCK[naf$c_length],          ! Place to put a record
;  969          1363  2     user_desc : VECTOR[2],                      ! Username descriptor
;  970          1364  2     ptr,                                        ! Temp pointer
;  971          1365  2     node_len,                                   ! Length of node
;  972          1366  2     node_ptr,                                   ! Pointer to beginning of node
;  973          1367  2     user_len,                                   ! Length of username
;  974          1368  2     user_ptr;                                   ! Pointer to beginning of username
;  975          1369  2
;  976          1370  2 !
;  977          1371  2 ! Initialize the FAB and RAB
;  978          1372  2 !
;  979        P 1373  2 $FAB_INIT ( FAB = netfab,
;  980        P 1374  2              FAC = get,                         ! Want to get records
;  981        P 1375  2              FNM = 'NETUAF',                    ! Name is NETUAF
;  982        P 1376  2              DNM = 'SYS$SYSTEM:.DAT',           ! Look in SYS$SYSTEM
;  983          1377  2              SHR = (get,put,upd,del));          ! Do shared access
;  984          1378  2 !
;  985          1379  2 ! Disable group and process logical name translation.  This must be
;  986          1380  2 ! done manually, since $FAB_INIT doesn't know about the disable mask.
;  987          1381  2 !
;  988          1382  2 netfab[fab$v_lnm_mode] = psl$c_exec;
;  989          1383  2
;  990        P 1384  2 $RAB_INIT ( RAB = netrab,
;  991        P 1385  2              ROP = rrl,                         ! Don't lock records
;  992        P 1386  2              RAC = key,                         ! Access is keyed
;  993        P 1387  2              KRF = 0,                           ! Use primary key
;  994        P 1388  2              KBF = net_record[naf$t_remname],   ! Lookup key overlays net record
;  995        P 1389  2              KSZ = naf$s_remname,               ! and it's this long
;  996        P 1390  2              UBF = net_record,                  ! Fetch record and put it here
```

```
;  997        P 1391  2              USZ = naf$c_length,                     ! Size of record
;  998          1392  2              FAB = netfab);
;  999          1393  2
; 1000          1394  2 !
; 1001          1395  2 ! Open NETUAF
; 1002          1396  2 !
; 1003          1397  2 set_sysprv ();
; 1004          1398  3 IF NOT (status = $OPEN (FAB = netfab))
; 1005          1399  2 THEN
; 1006          1400  3     BEGIN
; 1007          1401  3     clear_sysprv ();
; 1008          1402  3     IF .status EQL rms$_fnf
; 1009          1403  3     THEN RETURN false;
; 1010          1404  3     SIGNAL_STOP (lgi$_netuafacc, 0, .status, .netfab[fab$l_stv]);
; 1011          1405  2     END;
; 1012          1406  2
; 1013          1407  3 IF NOT (status = $CONNECT (RAB = netrab))
; 1014          1408  2 THEN
; 1015          1409  3     BEGIN
; 1016          1410  3     IF .status EQL rms$_crmp
; 1017          1411  3     THEN
; 1018          1412  4         BEGIN
; 1019          1413  4         netfab[fab$w_gbc] = 0;
; 1020          1414  4         status = $CONNECT (RAB = netrab);
; 1021          1415  3         END;
; 1022          1416  3     IF NOT .status
; 1023          1417  3     THEN
; 1024          1418  4         BEGIN
; 1025          1419  4         clear_sysprv ();
; 1026          1420  4         $CLOSE (FAB = netfab);
; 1027          1421  4         SIGNAL_STOP (lgi$_netuafacc, 0, .status, .netrab[rab$l_stv]);
; 1028          1422  3         END;
; 1029          1423  2     END;
; 1030          1424  2
; 1031          1425  2 clear_sysprv ();
; 1032          1426  2
; 1033          1427  2 !
; 1034          1428  2 ! Get the remote node and remote username from the Network Control Block.
; 1035          1429  2 ! The NCB is an ASCII string that looks like this:
; 1036          1430  2 !
; 1037          1431  2 !       NODE::"OBJECT=USERNAME/<more stuff>"
; 1038          1432  2 !
; 1039          1433  2 ! Where NODE and USERNAME are the two fields to extract and use as a key,
; 1040          1434  2 ! to locate the record in NETUAF.DAT which contains the local username to
; 1041          1435  2 ! map to.
; 1042          1436  2 !
; 1043          1437  2
; 1044          1438  2 !
; 1045          1439  2 ! First, get the node.
; 1046          1440  2 !
; 1047          1441  2
; 1048          1442  2 ptr = CH$FIND_SUB ( .sys$error[0],                ! Search the NCB string
; 1049          1443  2                    .sys$error[1],
; 1050          1444  2                    2, UPLIT ('::'));           ! Looking for ::
; 1051          1445  2
; 1052          1446  2 !
; 1053          1447  2 ! If the node wasn't there, then return FALSE and process with no proxy
```

```
;  1054          1448  2 !
;  1055          1449  2
;  1056          1450  2 IF   .ptr EQL 0 OR
;  1057          1451  2      .ptr EQL .sys$error[1]
;  1058          1452  2 THEN RETURN false;
;  1059          1453  2
;  1060          1454  2 node_len = .ptr - .sys$error[1];                ! Store node length
;  1061          1455  2 node_ptr = .sys$error[1];                       ! And starting address
;  1062          1456  2
;  1063          1457  2 !
;  1064          1458  2 ! Get the username.  This is done by looking for the "=", then the
;  1065          1459  2 ! "/", and interpreting whatever is between the two characters as the
;  1066          1460  2 ! username.
;  1067          1461  2 !
;  1068          1462  2
;  1069          1463  2 ptr = CH$FIND_CH ( .sys$error[0],                ! Search the NCB string
;  1070          1464  2                    .sys$error[1],
;  1071          1465  2                    '=');                        ! Looking for equal sign
;  1072          1466  2
;  1073          1467  2 IF .ptr EQL 0                                    ! If no such character found
;  1074          1468  2 THEN RETURN false;                              ! return a value of FALSE
;  1075          1469  2
;  1076          1470  2 user_ptr = .ptr + 1;                            ! Compute beginning of username
;  1077          1471  2
;  1078          1472  2 ptr = CH$FIND_CH ( .sys$error[0],                ! Search the NCB string
;  1079          1473  2                    .sys$error[1],
;  1080          1474  2                    '/');                        ! Looking for slash
;  1081          1475  2
;  1082          1476  2 !
;  1083          1477  2 ! If no slash, or a null username, return FALSE
;  1084          1478  2 !
;  1085          1479  2
;  1086          1480  2 IF   .ptr EQL 0 OR
;  1087          1481  2      .ptr EQL .user_ptr
;  1088          1482  2 THEN RETURN false;
;  1089          1483  2
;  1090          1484  2 !
;  1091          1485  2 ! Otherwise, compute the username length
;  1092          1486  2 !
;  1093          1487  2
;  1094          1488  2 user_len = .ptr - .user_ptr;
;  1095          1489  2
;  1096          1490  2 !
;  1097          1491  2 ! Copy the node and username to NET_KEY, the key buffer that RMS will
;  1098          1492  2 ! use to look for the specified record.
;  1099          1493  2 !
;  1100          1494  2 CH$COPY ( .node_len, .node_ptr,                  ! Copy the nodename
;  1101          1495  2                                                  ! Padded with blanks
;  1102          1496  2           naf$s_node, net_record[naf$t_node]);   ! To the key buffer
;  1103          1497  2
;  1104          1498  2 CH$COPY ( .user_len, .user_ptr,                  ! Copy the username
;  1105          1499  2                                                  ! Padded with blanks
;  1106          1500  2           naf$s_remuser, net_record[naf$t_remuser]);
;  1107          1501  2
;  1108          1502  2 !
;  1109          1503  2 ! Now perform a $GET, so see if there is a record in NETUAF that
;  1110          1504  2 ! exactly matches the node and username specified.  If no exact match
```

```
;  1111          1505  2 ! is found, wildcarding is applied in the following order:
;  1112          1506  2 !
;  1113          1507  2 !       Wildcard node, specific user
;  1114          1508  2 !       Specific node, wildcard user
;  1115          1509  2 !       Wildcard node, wildcard user
;  1116          1510  2 !
;  1117          1511  2 ! If a match is found, then it is used and no further checking is done.
;  1118          1512  2 !
;  1119          1513  3 IF NOT ($GET (RAB = netrab))
;  1120          1514  2 THEN
;  1121          1515  3     BEGIN
;  1122          1516  3     CH$COPY ( 1, UPLIT ('*'),                   ! Put in wildcard node
;  1123          1517  3               naf$s_node, net_record[naf$t_node]);
;  1124          1518  3     IF NOT ($GET (RAB = netrab))
;  1125          1519  4     THEN
;  1126          1520  3         BEGIN
;  1127          1521  4         CH$COPY ( .node_len, .node_ptr,          ! Specific node,
;  1128          1522  4                   naf$s_node, net_record[naf$t_node]);
;  1129          1523  4         CH$COPY ( 1, UPLIT ('*'),                ! Wildcard user
;  1130          1524  4                   naf$s_remuser, net_record[naf$t_remuser]);
;  1131          1525  4         IF NOT ($GET (RAB = netrab))
;  1132          1526  4         THEN
;  1133          1527  4             BEGIN
;  1134          1528  5             CH$COPY ( 1, UPLIT ('*'),            ! Wildcard node and user
;  1135          1529  4                       naf$s_node, net_record[naf$t_node]);
;  1136          1530  5             IF NOT ($GET (RAB = netrab))
;  1137          1531  5             THEN
;  1138          1532  5                 BEGIN
;  1139          1533  5                 $CLOSE(FAB = netfab);
;  1140          1534  6                 RETURN false;                    ! If no matches, return false
;  1141          1535  5                 END;
;  1142          1536  6             END;
;  1143          1537  6         END;
;  1144          1538  6     END;
;  1145          1539  5 !
;  1146          1540  4 ! Close NETUAF
;  1147          1541  3 !
;  1148          1542  2 $CLOSE (FAB = netfab);
;  1149          1543  2 !
;  1150          1544  2 ! If we get here, then a match was found.  Check to see if the local username
;  1151          1545  2 ! is actually a "*", in which case copy the remote username to the local
;  1152          1546  2 ! username.
;  1153          1547  2 !
;  1154          1548  2 IF .VECTOR [net_record[naf$t_localuser], 0; ,BYTE] EQL '*'
;  1155          1549  2 THEN CH$COPY (.user_len, .user_ptr,
;  1156          1550  2                naf$s_localuser, net_record[naf$t_localuser]);
;  1157          1551  2 !
;  1158          1552  2 ! Now fill in the user descriptor with the local username, and call
;  1159          1553  2 ! GET_UAFREC, to get the UAF record without checking for password.
;  1160          1554  2 !
;  1161          1555  2 !
;  1162          1556  2 !
;  1163          1557  2 !
;  1164          1558  2 !
;  1165          1559  2 !
;  1166          1560  2 !
;  1167          1561  2 !
```

```
;  1168          1562  2 !
;  1169          1563  2
;  1170          1564  2 user_desc[0] = naf$s_localuser;
;  1171          1565  2 user_desc[1] = net_record[naf$t_localuser];
;  1172          1566  2
;  1173          1567  2 get_uafrec (user_desc);
;  1174          1568  2
;  1175          1569  2 !
;  1176          1570  2 ! Done.  If a UAF record was found, return TRUE.  Otherwise return FALSE.
;  1177          1571  2 !
;  1178          1572  2
;  1179          1573  2 RETURN (.uaf_record NEQ 0);
;  1180          1574  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  0005B P.AAM:  .ASCII  \NETUAF\                                        ;
                                                  00061 P.AAN:  .ASCII  \SYS$SYSTEM:.DAT\                               ;
                                                  00070 P.AAO:  .ASCII  \::\<0><0>                                      ;
                                                  00074 P.AAP:  .ASCII  \*\<0><0><0>                                    ;
                                                  00078 P.AAQ:  .ASCII  \*\<0><0><0>                                    ;
                                                  0007C P.AAR:  .ASCII  \*\<0><0><0>                                    ;

                                                                .EXTRN  SYS$OPEN, SYS$CONNECT
                                                                .EXTRN  SYS$CLOSE, SYS$GET

                                                                .PSECT  $CODE$,NOWRT,2

                                                  00000 GET_PROXY:
                                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11            ; 1334
                                                  00002         MOVAB   CLEAR_SYSPRV, R11                               ;
                                                  00009         MOVAB   SYS$GET, R10                                    ;
                                                  00010         MOVAB   -256(SP), SP                                    ;
0050  8F      00                                  00015         MOVC5   #0, (SP), #0, #80, $RMS_PTR                     ; 1377
                                                  0001C
                                                  0001E         MOVW    #20483, $RMS_PTR                                ;
                                                  00024         MOVW    #3842, $RMS_PTR+22                              ;
                                                  0002A         MOVB    #2, $RMS_PTR+31                                 ;
                                                  0002E         MOVAB   P.AAM, $RMS_PTR+44                              ;
                                                  00034         MOVAB   P.AAN, $RMS_PTR+48                              ;
                                                  0003A         MOVW    #3846, $RMS_PTR+52                              ;
  FA  AD      02                                  00040         INSV    #1, #0, #2, NETFAB+74                           ; 1382
0044  8F      00                                  00046         MOVC5   #0, (SP), #0, #68, $RMS_PTR                     ; 1392
                                                  0004D
                                                  0004F         MOVW    #17409, $RMS_PTR                                ;
                                                  00055         MOVL    #8, $RMS_PTR+4                                  ;
                                                  00059         MOVB    #1, $RMS_PTR+30                                 ;
                                                  0005D         MOVZBW  #100, $RMS_PTR+32                               ;
                                                  00062         MOVAB   NET_RECORD, $RMS_PTR+36                         ;
                                                  00067         MOVAB   NET_RECORD, $RMS_PTR+48                         ;
                                                  0006C         MOVB    #64, $RMS_PTR+52                                ;
                                                  00071         MOVAB   NETFAB, $RMS_PTR+60                             ;
                                  00000000G   00  00076         CALLS   #0, SET_SYSPRV                                  ; 1397
                                                  0007D         PUSHAB  NETFAB                                          ; 1398
                                  00000000G   00  00080         CALLS   #1, SYS$OPEN                                    ;
```

```
                52              50  D0 00087          MOVL    R0, STATUS                                      ;
                23              52  E8 0008A          BLBS    STATUS, 2$                                      ;
                6B              00  FB 0008D          CALLS   #0, CLEAR_SYSPRV                                ; 1401
00018292        8F              52  D1 00090          CMPL    STATUS, #98962                                  ; 1402
                                03  12 00097          BNEQ    1$                                              ;
                            014B  31 00099          BRW     12$                                             ;
                        BC      AD  DD 0009C 1$:      PUSHL   NETFAB+12                                       ; 1404
                                52  DD 0009F          PUSHL   STATUS                                          ;
                                7E  D4 000A1          CLRL    -(SP)                                           ;
                00000000G       8F  DD 000A3          PUSHL   #LGI$_NETUAFACC                                 ;
00000000G       00              04  FB 000A9          CALLS   #4, LIB$STOP                                    ;
                        6C      AE  9F 000B0 2$:      PUSHAB  NETRAB                                          ; 1407
00000000G       00              01  FB 000B3          CALLS   #1, SYS$CONNECT                                 ;
                52              50  D0 000BA          MOVL    R0, STATUS                                      ;
                3D              52  E8 000BD          BLBS    STATUS, 4$                                      ;
0001C14C        8F              52  D1 000C0          CMPL    STATUS, #115020                                 ; 1410
                                10  12 000C7          BNEQ    3$                                              ;
                        F8      AD  B4 000C9          CLRW    NETFAB+72                                       ; 1413
                        6C      AE  9F 000CC          PUSHAB  NETRAB                                          ; 1414
00000000G       00              01  FB 000CF          CALLS   #1, SYS$CONNECT                                 ;
                52              50  D0 000D6          MOVL    R0, STATUS                                      ;
                21              52  E8 000D9 3$:      BLBS    STATUS, 4$                                      ; 1416
                6B              00  FB 000DC          CALLS   #0, CLEAR_SYSPRV                                ; 1419
                        B0      AD  9F 000DF          PUSHAB  NETFAB                                          ; 1420
00000000G       00              01  FB 000E2          CALLS   #1, SYS$CLOSE                                   ;
                        78      AE  DD 000E9          PUSHL   NETRAB+12                                       ; 1421
                                52  DD 000EC          PUSHL   STATUS                                          ;
                                7E  D4 000EE          CLRL    -(SP)                                           ;
                00000000G       8F  DD 000F0          PUSHL   #LGI$_NETUAFACC                                 ;
                    00000000G   00  04  FB 000F6      CALLS   #4, LIB$STOP                                    ;
                                6B  00  FB 000FD 4$:  CALLS   #0, CLEAR_SYSPRV                                ; 1425
                            55  00000000G  00  D0 00100  MOVL    SYS$ERROR, R5                                   ; 1442
                            54  00000000G  00  D0 00107  MOVL    SYS$ERROR+4, R4                                 ; 1443
64              55      0000'   CF          02  39 0010E  MATCHC  #2, P.AAO, R5, (R4)                          ; 1444
                                            03  13 00115  BEQL   5$                                             ;
                            53              02  D0 00117  MOVL    #2, R3                                          ;
                            53              02  C2 0011A 5$:  SUBL2   #2, R3                                  ;
                                            2D  13 0011D  BEQL   8$                                             ; 1450
                            54              53  D1 0011F  CMPL    PTR, R4                                         ; 1451
                                            28  13 00122  BEQL   8$                                             ;
                57              53          54  C3 00124  SUBL3   R4, PTR, NODE_LEN                            ; 1454
                            56              54  D0 00128  MOVL    R4, NODE_PTR                                    ; 1455
                64          55              3D  3A 0012B  LOCC    #61, R5, (R4)                                   ; 1463
                                            02  12 0012F  BNEQ   6$                                             ;
                                            51  D4 00131  CLRL    R1                                              ;
                            53              51  D0 00133 6$:  MOVL    R1, PTR                                         ;
                                            78  13 00136  BEQL   9$                                             ; 1467
                            58      01      A3  9E 00138  MOVAB   1(R3), USER_PTR                                 ; 1470
                64          55              2F  3A 0013C  LOCC    #47, R5, (R4)                                   ; 1472
                                            02  12 00140  BNEQ   7$                                             ;
                                            51  D4 00142  CLRL    R1                                              ;
                            53              51  D0 00144 7$:  MOVL    R1, PTR                                         ;
                                            67  13 00147  BEQL   9$                                             ; 1480
                            58              53  D1 00149  CMPL    PTR, USER_PTR                                   ; 1481
                                            62  13 0014C 8$:  BEQL   9$                                             ;
                59          53              58  C3 0014E  SUBL3   USER_PTR, PTR, USER_LEN                       ; 1488
20              20          66              57  2C 00152  MOVC5   NODE_LEN, (NODE_PTR), #32, #32, NET_RECORD    ; 1496
```

```
                                      08  AE      00157
     20        20          68             59  2C 00159         MOVC5   USER_LEN, (USER_PTR), #32, #32, -          ; 1500
                                      28  AE      0015E                NET_RECORD+32
                                      6C  AE  9F 00160         PUSHAB  NETRAB                                     ; 1513
                           6A             01  FB 00163         CALLS   #1, SYS$GET
                           49             50  E8 00166         BLBS    R0, 10$
     20        20    0000' CF             01  2C 00169         MOVC5   #1, P.AAP, #32, #32, NET_RECORD            ; 1518
                                      08  AE      00170
                                      6C  AE  9F 00172         PUSHAB  NETRAB                                     ; 1519
                           6A             01  FB 00175         CALLS   #1, SYS$GET
                           37             50  E8 00178         BLBS    R0, 10$
     20        20          66             57  2C 0017B         MOVC5   NODE_LEN, (NODE_PTR), #32, #32, NET_RECORD ; 1524
                                      08  AE      00180
     20        20    0000' CF             01  2C 00182         MOVC5   #1, P.AAQ, #32, #32, NET_RECORD+32         ; 1527
                                      28  AE      00189
                                      6C  AE  9F 0018B         PUSHAB  NETRAB                                     ; 1528
                           6A             01  FB 0018E         CALLS   #1, SYS$GET
                           1E             50  E8 00191         BLBS    R0, 10$
     20        20    0000' CF             01  2C 00194         MOVC5   #1, P.AAR, #32, #32, NET_RECORD            ; 1533
                                      08  AE      0019B
                                      6C  AE  9F 0019D         PUSHAB  NETRAB                                     ; 1534
                           6A             01  FB 001A0         CALLS   #1, SYS$GET
                           0C             50  E8 001A3         BLBS    R0, 10$
                                      B0  AD  9F 001A6         PUSHAB  NETFAB                                     ; 1537
                 00000000G 00             01  FB 001A9         CALLS   #1, SYS$CLOSE
                                          35  11 001B0 9$:     BRB     12$                                        ; 1538
                                      B0  AD  9F 001B2 10$:    PUSHAB  NETFAB                                     ; 1547
                 00000000G 00             01  FB 001B5         CALLS   #1, SYS$CLOSE
                           2A         48  AE  91 001BC         CMPB    NET_RECORD+64, #42                         ; 1554
                                          07  12 001C0         BNEQ    11$
     20        20          68             59  2C 001C2         MOVC5   USER_LEN, (USER_PTR), #32, #32, -          ; 1557
                                      48  AE      001C7                NET_RECORD+64
                           6E             20  D0 001C9 11$:    MOVL    #32, USER_DESC                             ; 1564
                      04   AE         48  AE  9E 001CC         MOVAB   NET_RECORD+64, USER_DESC+4                 ; 1565
                                          5E  DD 001D1         PUSHL   SP                                         ; 1567
                 00000000G 00             01  FB 001D3         CALLS   #1, GET_UAFREC
                                          50  D4 001DA         CLRL    R0                                         ; 1573
                               00000000G  00  D5 001DC         TSTL    UAF_RECORD
                                          05  13 001E2         BEQL    13$
                                          50  D6 001E4         INCL    R0
                                              04 001E6         RET
                                          50  D4 001E7 12$:    CLRL    R0                                         ; 1574
                                              04 001E9 13$:    RET
```

; Routine Size: 490 bytes,   Routine Base: $CODE$ + 0625

```
; 1182          1575  1 END
; 1183          1576  0 ELUDOM


                                                .EXTRN  LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  $OWN$                         299  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                        128  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       2063  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics
;
;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1      18619       163         0      1000          00:01.4


;                                       COMMAND QUALIFIERS

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

; Size:          2063 code + 427 data bytes
; Run Time:         00:31.3
; Elapsed Time:     02:05.2
; Lines/CPU Min:     3020
; Lexemes/CPU-Min: 38117
; Memory Used:  286 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PPDDEF
MDL

FILEIO
LIS

LOGIN

DETACHED
LIS

LOGINOUT
MAP

UTILDEF
REQ

LOGINCMD
CLD

HPWD
LIS

LGIDEF
MDL

AUDIT
LIS

BREAKIN
LIS