```
DDDDDDDDDDD         IIIIIIIII     FFFFFFFFFFFFFFF
DDDDDDDDDDDD        IIIIIIIII     FFFFFFFFFFFFFFF
DDDDDDDDDDDD        IIIIIIIII     FFFFFFFFFFFFFFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFFFFFFFFFFF
DDD            DDD     III       FFFFFFFFFFFF
DDD            DDD     III       FFFFFFFFFFFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDD            DDD     III       FFF
DDDDDDDDDDDD        IIIIIIIII     FFF
DDDDDDDDDDDD        IIIIIIIII     FFF
DDDDDDDDDDDD        IIIIIIIII     FFF
```

```
MM        MM     AAAAAA       IIIIII     NN        NN
MM        MM     AAAAAA       IIIIII     NN        NN
MMMM    MMMM   AA      AA       II       NN        NN
MMMM    MMMM   AA      AA       II       NN        NN
MM  MM  MM  MM AA      AA       II       NNNN      NN
MM  MM  MM  MM AA      AA       II       NNNN      NN
MM      MM  MM AA      AA       II       NN  NN    NN
MM      MM  MM AA      AA       II       NN  NN    NN
MM      MM  MM AAAAAAAAAA       II       NN    NNNN
MM      MM  MM AAAAAAAAAA       II       NN    NNNN
MM      MM  MM AA      AA       II       NN      NN      ....
MM      MM  MM AA      AA       II       NN      NN      ....
MM      MM  MM AA      AA    IIIIII      NN      NN      ....
MM      MM  MM AA      AA    IIIIII      NN      NN      ....


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 DIF_MAIN (                        ! Differences main routine
;     2        0002  0                 LANGUAGE (BLISS32),
;     3        0003  0                 ADDRESSING_MODE (EXTERNAL=GENERAL,
;     4        0004  0                                  NONEXTERNAL=LONG_RELATIVE),
;     5        0005  0                 MAIN = DIF$START,
;     6        0006  0                 IDENT = 'V04-000'
;     7        0007  0                 ) =
;     8        0008  1 BEGIN
;     9        0009  1 !*****************************************************************************
;    10        0010  1 !*                                                                           *
;    11        0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    12        0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    13        0013  1 !*  ALL RIGHTS RESERVED.                                                      *
;    14        0014  1 !*                                                                           *
;    15        0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    16        0016  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE          *
;    17        0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER     *
;    18        0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    19        0019  1 !*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY       *
;    20        0020  1 !*  TRANSFERRED.                                                              *
;    21        0021  1 !*                                                                           *
;    22        0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
;    23        0023  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
;    24        0024  1 !*  CORPORATION.                                                              *
;    25        0025  1 !*                                                                           *
;    26        0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS     *
;    27        0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    28        0028  1 !*                                                                           *
;    29        0029  1 !*                                                                           *
;    30        0030  1 !*****************************************************************************
;    31        0031  1
;    32        0032  1 !++
;    33        0033  1 !
;    34        0034  1 ! FACILITY:     DCL Differences command
;    35        0035  1 !
;    36        0036  1 ! ABSTRACT:
;    37        0037  1 !                The DCL DIFFERENCES command compares the contents of
;    38        0038  1 !                two files.
;    39        0039  1 !
;    40        0040  1 ! ENVIRONMENT:
;    41        0041  1 !                VAX native, user mode
;    42        0042  1 !
;    43        0043  1 !--
;    44        0044  1 !
;    45        0045  1 ! AUTHOR:       Peter George, Benn Schreiber    CREATION DATE:  1-August-1981
;    46        0046  1 !
;    47        0047  1 ! MODIFIED BY:
;    48        0048  1 !
;    49        0049  1 !       V03-003 PCG0005         Peter George            13-Oct-1983
;    50        0050  1 !               Fix bugs in maximum differences logic.
;    51        0051  1 !               Let image rundown clean up VM usage.
;    52        0052  1 !
;    53        0053  1 !       V03-002 BLS0212         Benn Schreiber          14-Mar-1983
;    54        0054  1 !               Correct handling of 0-length records which caused problem
;    55        0055  1 !               with /ignore=exact.  Set rdb$v_edited if tabs converted to
;    56        0056  1 !               blanks.
;    57        0057  1 !
```

```
;    58          0058  1 !       V03-001 PCG0004         Peter George            05-Jan-1983
;    59          0059  1 !               Clean up some code.  Speed up record editing code.
;    60          0060  1 !--
;    61          0061  1
```

```
;    62          0062  1 LIBRARY
;    63          0063  1     'SYS$LIBRARY:STARLET.L32';
;    64          0064  1
;    65          0065  1 REQUIRE
;    66          0066  1         'DIFPRE';                          ! DIF prefix file
;    67          0140  1 REQUIRE
;    68          0141  1         'DIFDEF';                          ! DIF data structures
;    69          0372  1
;    70          0373  1 !
;    71          0374  1 ! Difference global data
;    72          0375  1 !
;    73          0376  1 EXTERNAL
;    74          0377  1     dif$gl_commdesc : BBLOCK,              ! Desc for buffer of comment delimiters
;    75          0378  1     dif$gl_commflgs : BITVECTOR,           ! Bit is set if corresponding char must be in first column
;    76          0379  1     dif$gl_ignore : BBLOCK,                ! Flags of characters to ignore
;    77          0380  1     dif$gl_cmdesc : BBLOCK,                ! Command line descriptor
;    78          0381  1     dif$gl_header,                         ! No. of lines to skip as header
;    79          0382  1     dif$gl_match,                          ! No. of records that constitute a match
;    80          0383  1     dif$gl_maxdif,                         ! Maximum number of unmatched records
;    81          0384  1     dif$gl_merged,                         ! No. of matched lines to follow each list of merged differe
;    82          0385  1     dif$gl_parallel,                       ! No. of matched lines to follow each list of parallel diffe
;    83          0386  1     dif$gl_wndwsiz,                        ! No. of records to search before declaring a mismatch
;    84          0387  1     dif$gl_flags : BBLOCK,                 ! Flags
;    85          0388  1     dif$gl_difrec,                         ! No. of different records detected
;    86          0389  1     dif$gl_difsec,                         ! No. of difference sections detected
;    87          0390  1     dif$gl_width,                          ! Width of lines in output listing
;    88          0391  1     dif$gl_inbuf,                          ! Address of the input record buffer
;    89          0392  1     dif$gl_outbsiz,                        ! Allocated size of output buffer
;    90          0393  1     dif$gl_outbuf,                         ! Address of the output record buffer
;    91          0394  1 !
;    92          0395  1 ! Input and output file data structures
;    93          0396  1 !
;    94          0397  1     dif$gl_masfdb : BBLOCK,                ! Master file fdb
;    95          0398  1     dif$gl_masrab : BBLOCK,                ! RAB for master file
;    96          0399  1     dif$gl_maseof : BBLOCK,                ! Master file EOF RDB
;    97          0400  1     dif$gl_revfdb : BBLOCK,                ! Revision file fdb
;    98          0401  1     dif$gl_revrab : BBLOCK,                ! RAB for revision file
;    99          0402  1     dif$gl_reveof : BBLOCK;                ! Revision file EOF RDB
;   100          0403  1
;   101          0404  1 EXTERNAL ROUTINE
;   102          0405  1     dif$getcmd,                            ! Initialize global data
;   103          0406  1     dif$open_mas,                          ! Open master input file
;   104          0407  1     dif$open_rev,                          ! Open revision input file
;   105          0408  1     dif$open_out,                          ! Open output file
;   106          0409  1     dif$close_in,                          ! Close input file
;   107          0410  1     dif$close_out,                         ! Close output file
;   108          0411  1     lib$get_vm,                            ! Allocate virtual memory
;   109          0412  1     lib$free_vm,                           ! Deallocate virtual memory
;   110          0413  1     sys$fao;                               ! FAO conversion routine
;   111          0414  1
;   112          0415  1 EXTERNAL ROUTINE
;   113          0416  1     additional_output,                     ! Output 2nd, 3rd, ... listings
;   114          0417  1     init_hex_octal,                        ! Prepare for hex or octal output
;   115          0418  1     output_listing_trailer,                ! Output listing trailer
;   116          0419  1     put_record,                            ! Output a record in appropriate radix
;   117          0420  1     write_mismatch;                        ! Output records in a mismatch
;   118          0421  1
```

```
;  120          0422  1 FORWARD ROUTINE
;  121          0423  1       allocate_rdb,                   ! Allocate an RDB
;  122          0424  1       compare,                        ! Compare two records
;  123          0425  1       get_record,                     ! Get a record from the list or file
;  124          0426  1       mark_match,                     ! Mark all matched records in list
;  125          0427  1       mismatch,                       ! Mismatch found, find next match
;  126          0428  1       print_and_quit,                 ! Print last records processed and quit
;  127          0429  1       process_record,                 ! Delete ignore chars from record
;  128          0430  1       purge_rdb,                      ! Purge processed RDB's
;  129          0431  1       read_record,                    ! Read a record from a file
;  130          0432  1       set_move_flags,                 ! Set fdb move flags
;  131          0433  1       test_match;                     ! Verify match contains enough records
;  132          0434  1
;  133          0435  1 EXTERNAL LITERAL
;  134          0436  1       dif$_filaredif,
;  135          0437  1       dif$_insvirmem,
;  136          0438  1       dif$_maxdif,
;  137          0439  1       dif$_readerr,
;  138          0440  1       dif$_samefile;
; R
```

```
;  140          0441  1 GLOBAL ROUTINE dif$start =
;  141          0442  2 BEGIN
;  142          0443  2
;  143          0444  2 !++
;  144          0445  2 !
;  145          0446  2 ! FUNCTIONAL DESCRIPTION:
;  146          0447  2 !
;  147          0448  2 !       This routine is called after the Difference command has been
;  148          0449  2 !       parsed by the CLI.  It calls appropriate initialization routines,
;  149          0450  2 !       and then starts the actual differences processing.  A simple
;  150          0451  2 !       loop is executed in which lines from each file are compared until
;  151          0452  2 !       either a mismatch is detected, or the end of both files has been
;  152          0453  2 !       reached.  On completion, routines are called to output any additional
;  153          0454  2 !       listings that have been requested and to close all open files.
;  154          0455  2 !
;  155          0456  2 ! INPUTS:
;  156          0457  2 !
;  157          0458  2 !       None
;  158          0459  2 !
;  159          0460  2 ! OUTPUTS:
;  160          0461  2 !
;  161          0462  2 !       The requested differences listing is generated.
;  162          0463  2 !
;  163          0464  2 ! IMPLICIT INPUTS:
;  164          0465  2 !
;  165          0466  2 !       The command line is parsed and the appropriate global symbols
;  166          0467  2 !       have been initialized.
;  167          0468  2 !
;  168          0469  2 ! ROUTINE VALUES:
;  169          0470  2 !
;  170          0471  2 !       Always true.
;  171          0472  2 !
;  172          0473  2 !--
;  173          0474  2 LOCAL
;  174          0475  2     inbufalloc,                 ! Flag indicating that input buffer was allocated
;  175          0476  2     masrdb : REF BBLOCK,        ! Temporary storage for a master file RDB address
;  176          0477  2     revrdb : REF BBLOCK,        ! Temporary storage for a revision file RDB address
;  177          0478  2     status;
;  178          0479  2
;  179          0480  2 !
;  180          0481  2 ! Call initialization routines.
;  181          0482  2 !
;  182          0483  2 dif$getcmd ();                  ! Initialize global data
;  183          0484  2 dif$open_mas ();                ! Open master input file
;  184          0485  2 dif$open_rev ();                ! Open revision input file
;  185          0486  2 dif$open_out ();                ! Open output file
;  186          0487  2 set_move_flags ();              ! Set FDB move flags
;  187          0488  2
;  188          0489  2 !
;  189          0490  2 ! Adjust value of /WIDTH to:
;  190          0491  2 !       Avoid problems with huge values
;  191          0492  2 !       Avoid problems with tiny values
;  192          0493  2 !       Prevent data truncation with /SLP output
;  193          0494  2 !
;  194          0495  2 dif$gl_width = MAXU (MINU (.dif$gl_width, 65535), dif$c_minlisiz);
;  195          0496  2 IF .dif$gl_flags [dif$v_slp]
;  196          0497  2 THEN
```

```
;  197     0498  2     dif$gl_width = MAXU (
;  198     0499  2         .dif$gl_width,                  ! /WIDTH value
;  199     0500  2         .dif$gl_masrab [rab$w_usz],     ! master file record size bound
;  200     0501  2         .dif$gl_revrab [rab$w_usz]);    ! revision file record size bound
;  201     0502  2
;  202     0503  2 !
;  203     0504  2 ! Allocate memory for the output buffer.  The size calculation is based on
;  204     0505  2 ! an examintion of all used of this buffer.  The calculation was complicated
;  205     0506  2 ! enough that it may not be completely accurate.
;  206     0507  2 !
;  207     0508  2 dif$gl_outbsiz = MAXU (
;  208     0509  2     .dif$gl_width +                     ! specified /WIDTH value plus
;  209     0510  2         20,                             !  pad
;  210     0511  2     12,                                 ! OUTPUT for stars
;  211     0512  2     dif$c_minlisiz,                     ! minimum size of listing boilerplate
;  212     0513  2     19+dif$c_entrysize-dif$c_linenum,   ! OUTPUT put_record_hex_octal routine
;  213     0514  2     (8 + 4*dif$c_linenum)/3,            ! OUTPUT output_parallel routine
;  214     0515  2     2*dif$c_linenum);                   ! OUTPUT output_slp routine
;  215     0516  3 IF NOT (status = LIB$GET_VM (dif$gl_outbsiz, dif$gl_outbuf))
;  216     0517  2     THEN SIGNAL_STOP (.status);
;  217     0518  2
;  218     0519  2 !
;  219     0520  2 ! Allocate memory for the input buffer, if required, and set the input
;  220     0521  2 ! buffer allocation flag appropriately.
;  221     0522  2 !
;  222     0523  2 IF .dif$gl_flags [dif$v_hex] OR .dif$gl_flags [dif$v_octal]
;  223     0524  3     THEN BEGIN
;  224     0525  3          inbufalloc = true;
;  225     0526  4          IF NOT (status = LIB$GET_VM (%REF (MAXU (.dif$gl_masrab [rab$w_usz],
;  226     0527  4                  .dif$gl_revrab [rab$w_usz])), dif$gl_inbuf))
;  227     0528  3              THEN SIGNAL_STOP (.status);
;  228     0529  3          END
;  229     0530  2     ELSE inbufalloc = false;
;  230     0531  2
;  231     0532  2 !
;  232     0533  2 ! If the first listing will be in hex or octal, then initialize the appropriate
;  233     0534  2 ! global data.
;  234     0535  2 !
;  235     0536  2 IF NOT .dif$gl_flags [dif$v_ascii]
;  236     0537  2     THEN init_hex_octal ();
;  237     0538  2
;  238     0539  2 !
;  239     0540  2 ! Set flag for output routine to output header.
;  240     0541  2 !
;  241     0542  2 dif$gl_flags [dif$v_init] = true;
;  242     0543  2
;  243     0544  2 !
;  244     0545  2 ! This is the main loop that drives the search for differerces.  It processes
;  245     0546  2 ! matches itself, and calls mismatch to process differences.
;  246     0547  2 !
;  247     0548  3 DO BEGIN
;  248     0549  3
;  249     0550  3 !
;  250     0551  3 ! For each input file, get the RDB for the next record.  If we run out of
;  251     0552  3 ! virtual memory, then set up the appropriate data so that we can die gracefully.
;  252     0553  3 !
;  253     0554  4 IF NOT (status = get_record (dif$gl_masfdb))
```

```
;  254          0555  4          OR NOT (status = get_record (dif$gl_revfdb))
;  255          0556  4      THEN BEGIN
;  256          0557  4          dif$gl_masfdb [fdb$v_move] = false;              ! So that we don't generate any more listings
;  257          0558  4          dif$gl_revfdb [fdb$v_move] = false;
;  258          0559  4          EXITLOOP;
;  259          0560  3          END;
;  260          0561  3
;  261          0562  3      !
;  262          0563  3      ! If we are only using an input file's records in one listing file, then
;  263          0564  3      ! purge those RDB's that we are permanently finished with.
;  264          0565  3      !
;  265          0566  3      IF NOT .dif$gl_masfdb [fdb$v_move]
;  266          0567  3          THEN purge_rdb (dif$gl_masfdb);
;  267          0568  3
;  268          0569  3      IF NOT .dif$gl_revfdb [fdb$v_move]
;  269          0570  3          THEN purge_rdb (dif$gl_revfdb);
;  270          0571  3
;  271          0572  3      !
;  272          0573  3      ! Compare the two records that we have just fetched.
;  273          0574  3      ! If they differ, init the FIRSTDIF and CURREC FDB fields and call mismatch
;  274          0575  3      ! to process the difference section.  If mismatch encounters too many differences
;  275          0576  3      ! or runs out of VM, then die gracefully.
;  276          0577  3      ! If they are the same, and if we are generating a change bar listing,
;  277          0578  3      ! then print the master file record.
;  278          0579  3      !
;  279          0580  3      IF NOT compare (.dif$gl_masfdb [fdb$l_currec], .dif$gl_revfdb [fdb$l_currec])    ! Compare the two records
;  280          0581  4          THEN BEGIN                                                                   ! They differ
;  281          0582  4              dif$gl_masfdb [fdb$l_firstdif] = .dif$gl_masfdb [fdb$l_currec];          ! Init ptrs to start of dif
;  282          0583  4              dif$gl_revfdb [fdb$l_firstdif] = .dif$gl_revfdb [fdb$l_currec];
;  283          0584  4              dif$gl_masfdb [fdb$l_comprec] = .dif$gl_masfdb [fdb$l_currec];           ! Init ptrs for first record
;  284          0585  4              dif$gl_revfdb [fdb$l_comprec] = .dif$gl_revfdb [fdb$l_currec];           !   try and match
;  285          0586  4              status = mismatch (.dif$gl_wndwsiz);                                     ! Call mismatch with window
;  286          0587  4              IF (.status EQL dif$_maxdif) OR                                          ! Check return status
;  287          0588  5                 (.status EQL dif$_insvirmem)
;  288          0589  5                 THEN BEGIN                                                           ! If some problem
;  289          0590  5                      dif$gl_masfdb [fdb$v_move] = false;                             ! Then die gracefully
;  290          0591  5                      dif$gl_revfdb [fdb$v_move] = false;
;  291          0592  5                      EXITLOOP;
;  292          0593  4                      END;
;  293          0594  4              masrdb = .dif$gl_masfdb [fdb$l_currec];                                  ! Othewise, point to matched
;  294          0595  4              revrdb = .dif$gl_revfdb [fdb$l_currec];
;  295          0596  4              END
;  296          0597  4
;  297          0598  4          ELSE BEGIN                                                                       ! If records are the
;  298          0599  4              IF NOT .dif$gl_flags [dif$v_merged] AND NOT .dif$gl_flags [dif$v_parallel]    !   currently output
;  299          0600  4                 AND NOT .dif$gl_flags [dif$v_separated]                                !   listing, then ou
;  300          0601  4                 THEN IF .dif$gl_masfdb [fdb$v_changebar]                               !   record.
;  301          0602  4                         THEN put_record (dif$gl_masfdb, 2)
;  302          0603  4                         ELSE IF .dif$gl_revfdb [fdb$v_changebar]
;  303          0604  4                                 THEN put_record (dif$gl_revfdb, 2);
;  304          0605  4              masrdb = .dif$gl_masfdb [fdb$l_currec];                                  ! Po    to matched records
;  305          0606  4              revrdb = .dif$gl_revfdb [fdb$l_currec];
;  306          0607  4              masrdb [rdb$v_match] = revrdb [rdb$v_match] = true;                      ! Indicate that they are mat
;  307          0608  3              END;
;  308          0609  3
;  309          0610  3  END
;  310          0611  2  UNTIL (.masrdb [rdb$v_eof]);                                                         ! Quit if end of both files
```

```
;   311          0612  2
;   312          0613  2 !
;   313          0614  2 ! Finish off current listing and then output any other listings required.
;   314          0615  2 !
;   315          0616  2 dif$gl_masfdb [fdb$l_firstdif] = .dif$gl_masfdb [fdb$l_firstrec];          ! Set up ptrs for additional
;   316          0617  2 dif$gl_revfdb [fdb$l_firstdif] = .dif$gl_revfdb [fdb$l_firstrec];                ; R
;   317          0618  2 dif$gl_masfdb [fdb$l_compnrec] = dif$gl_maseof;
;   318          0619  2 dif$gl_revfdb [fdb$l_compnrec] = dif$gl_reveof;
;   319          0620  2 dif$gl_masfdb [fdb$l_lastrfa] = 0;
;   320          0621  2 dif$gl_revfdb [fdb$l_lastrfa] = 0;
;   321          0622  2 additional_output ();                                                        ! Call output routine
;   322          0623  2
;   323          0624  2 !
;   324          0625  2 ! Determine completion status.  Use worst possible.
;   325          0626  2 !
;   326          0627  3 IF (.status NEQ dif$_maxdif) AND (.status NEQ dif$_insvirmem)
;   327          0628  4    THEN (IF (.dif$gl_difsec EQL 0)
;   328          0629  3             THEN status = dif$_samefile
;   329          0630  2             ELSE status = dif$_filaredif);
;   330          0631  2
;   331          0632  2 !
;   332          0633  2 ! Output information at bottom of listing.
;   333          0634  2 !
;   334          0635  2 output_listing_trailer ();
;   335          0636  2
;   336          0637  2 !
;   337          0638  2 ! Close open files.
;   338          0639  2 !
;   339          0640  2 dif$close_in (dif$gl_masfdb);                  ! Close master input file
;   340          0641  2 dif$close_in (dif$gl_revfdb);                  ! Close revision input file
;   341          0642  2 dif$close_out ();                              ! Close output file
;   342          0643  2
;   343          0644  2 RETURN .status;
;   344          0645  1 END;                                           ! Of main


                .TITLE  DIF_MAIN
                .IDENT  \V04-000\

                .EXTRN  DIF$GL_COMMDESC
                .EXTRN  DIF$GL_COMMFLGS
                .EXTRN  DIF$GL_IGNORE, DIF$GL_CMDESC
                .EXTRN  DIF$GL_HEADER, DIF$GL_MATCH
                .EXTRN  DIF$GL_MAXDIF, DIF$GL_MERGED
                .EXTRN  DIF$GL_PARALLEL
                .EXTRN  DIF$GL_WNDWSIZ, DIF$GL_FLAGS
                .EXTRN  DIF$GL_DIFREC, DIF$GL_DIFSEC
                .EXTRN  DIF$GL_WIDTH, DIF$GL_INBUF
                .EXTRN  DIF$GL_OUTBSIZ, DIF$GL_OUTBUF
                .EXTRN  DIF$GL_MASFDB, DIF$GL_MASRAB
                .EXTRN  DIF$GL_MASEOF, DIF$GL_REVFDB
                .EXTRN  DIF$GL_REVRAB, DIF$GL_REVEOF
                .EXTRN  DIF$GETCMD, DIF$OPEN_MAS
                .EXTRN  DIF$OPEN_REV, DIF$OPEN_OUT
                .EXTRN  DIF$CLOSE_IN, DIF$CLOSE_OUT
                .EXTRN  LIB$GET_VM, LIB$FREE_VM
                .EXTRN  SYS$FAO, ADDITIONAL_OUTPUT
```

```
                                        .EXTRN  INIT_HEX_OCTAL, OUTPUT_LISTING_TRAILER
                                        .EXTRN  PUT_RECORD, WRITE_MISMATCH
                                        .EXTRN  DIF$_FILAREDIF, DIF$_INSVIRMEM
                                        .EXTRN  DIF$_MAXDIF, DIF$_READERR
                                        .EXTRN  DIF$_SAMEFILE

                                        .PSECT  $CODE$,NOWRT,2

                            0FFC 00000  .ENTRY  DIF$START, Save R2,R3,R4,R5,R6,R7,R8,R9,-    ; 0441
                                                R10,R11
                    5B 00000000G   00    9E 00002  MOVAB   DIF$GL_OUTBSIZ, R11
                    5A 00000000G   00    9E 00009  MOVAB   DIF$GL_MASRAB+32, R10
                    59 00000000G   00    9E 00010  MOVAB   DIF$GL_REVRAB+32, R9
                    58 00000000G   00    9E 00017  MOVAB   DIF$GL_WIDTH, R8
                    57 00000000G   00    9E 0001E  MOVAB   DIF$GL_FLAGS, R7
                    56 00000000G   00    9E 00025  MOVAB   DIF$GL_REVFDB, R6
                    55 00000000G   00    9E 0002C  MOVAB   DIF$GL_MASFDB, R5
                    5E             04    C2 00033  SUBL2   #4, SP
          00000000G 00             00    FB 00036  CALLS   #0, DIF$GETCMD                  ; 0483
          00000000G 00             00    FB 0003D  CALLS   #0, DIF$OPEN_MAS                ; 0484
          00000000G 00             00    FB 00044  CALLS   #0, DIF$OPEN_REV                ; 0485
          00000000G 00             00    FB 0004B  CALLS   #0, DIF$OPEN_OUT                ; 0486
          00000000V EF             00    FB 00052  CALLS   #0, SET_MOVE_FLAGS              ; 0487
                    50             68    D0 00059  MOVL    DIF$GL_WIDTH, R0                ; 0495
          0000FFFF  8F                   50  D1 0005C  CMPL    R0, #65535
                                         05  1B 00063  BLEQU   1$
                    50       FFFF        8F  3C 00065  MOVZWL  #65535, R0
                    0C                   50  D1 0006A 1$:  CMPL    R0, #12
                                         03  1E 0006D  BGEQU   2$
                    50                   0C  D0 0006F  MOVL    #12, R0
                    68                   50  D0 00072 2$:  MOVL    R0, DIF$GL_WIDTH
                    1A             01    A7  E9 00075  BLBC    DIF$GL_FLAGS+1, 5$          ; 0496
                    50                   68  D0 00079  MOVL    DIF$GL_WIDTH, R0            ; 0500
          50        6A                   10  00 ED 0007C  CMPZV   #0, #16, DIF$GL_MASRAB+32, R0
                                         03  1B 00081  BLEQU   3$
                    50                   6A  3C 00083  MOVZWL  DIF$GL_MASRAB+32, R0
          50        69                   10  00 ED 00086 3$:  CMPZV   #0, #16, DIF$GL_REVRAB+32, R0  ; 0501
                                         03  1B 0008B  BLEQU   4$
                    50                   69  3C 0008D  MOVZWL  DIF$GL_REVRAB+32, R0
                    68                   50  D0 00090 4$:  MOVL    R0, DIF$GL_WIDTH            ; 0498
                    50             68    14  C1 00093 5$:  ADDL3   #20, DIF$GL_WIDTH, R0      ; 0509
                    11                   50  D1 00097  CMPL    R0, #17                     ; 0508
                                         03  1E 0009A  BGEQU   6$
                    50                   11  D0 0009C  MOVL    #17, R0
                    6B                   50  D0 0009F 6$:  MOVL    R0, DIF$GL_OUTBSIZ
                         00000000G       00  9F 000A2  PUSHAB  DIF$GL_OUTBUF               ; 0516
                                         5B  DD 000A8  PUSHL   R11
                00000000G                00  FB 000AA  CALLS   #2, LIB$GET_VM
                    53                   50  D0 000B1  MOVL    R0, STATUS
                    09                   53  E8 000B4  BLBS    STATUS, 7$
                                         53  DD 000B7  PUSHL   STATUS                      ; 0517
                00000000G                00  FB 000B9  CALLS   #1, LIB$STOP
          04        67                   01  E0 000C0 7$:  BBS     #1, DIF$GL_FLAGS, 8$      ; 0523
          32        67                   02  E1 000C4  BBC     #2, DIF$GL_FLAGS, 10$
                    50                   01  D0 000C8 8$:  MOVL    #1, INBUFALLOC            ; 0525
                         00000000G       00  9F 000CB  PUSHAB  DIF$GL_INBUF               ; 0526
                       04        AE            6A  3C 000D1  MOVZWL  DIF$GL_MASRAB+32, 4(SP)  ; 0527
```

```
                04   AE          69  B1 000D5          CMPW    DIF$GL_REVRAB+32, 4(SP)                                 ;
                                 04  1B 000D9          BLEQU   9$                                                      ;
                04   AE          69  3C 000DB          MOVZWL  DIF$GL_REVRAB+32, 4(SP)                                 ;
                              04 AE  9F 000DF 9$:       PUSHAB  4(SP)                                                   ;
            00000000G         00     02  FB 000E2      CALLS   #2, LIB$GET_VM                                          ;
                              53     50  D0 000E9      MOVL    R0, STATUS                                              ;
                              0D     53  E8 000EC      BLBS    STATUS, 11$                                             ;
                                     53  DD 000EF      PUSHL   STATUS                                                  ;
            00000000G         00     01  FB 000F1      CALLS   #1, LIB$STOP                                            ;
                                     02  11 000F8      BRB     11$                                                     ;
                                     50  D4 000FA 10$:  CLRL    INBUFALLOC                                              ;
                              07     67  E8 000FC 11$:  BLBS    DIF$GL_FLAGS, 12$                                       ;
            00000000G         00     00  FB 000FF      CALLS   #0, INIT_HEX_OCTAL                                      ;
                01   A7              20  88 00106 12$:  BISB2   #32, DIF$GL_FLAGS+1                                    ;
                                     55  DD 0010A 13$:  PUSHL   R5                                                     ;
            00000000V         EF     01  FB 0010C      CALLS   #1, GET_RECORD                                          ;
                              53     50  D0 00113      MOVL    R0, STATUS                                              ;
                              6B     53  E9 00116      BLBC    STATUS, 16$                                             ;
                                     56  DD 00119      PUSHL   R6                                                      ;
            00000000V         EF     01  FB 0011B      CALLS   #1, GET_RECORD                                          ;
                              53     50  D0 00122      MOVL    R0, STATUS                                              ;
                              5C     53  E9 00125      BLBC    STATUS, 16$                                             ;
09              24   A5              03  E0 00128      BBS     #3, DIF$GL_MASFDB+36, 14$                               ;
                                     55  DD 0012D      PUSHL   R5                                                      ;
            00000000V         EF     01  FB 0012F      CALLS   #1, PURGE_RDB                                           ;
09              24   A6              03  E0 00136 14$:  BBS     #3, DIF$GL_REVFDB+36, 15$                               ;
                                     56  DD 0013B      PUSHL   R6                                                      ;
            00000000V         EF     01  FB 0013D      CALLS   #1, PURGE_RDB                                           ;
                                     66  DD 00144 15$:  PUSHL   DIF$GL_REVFDB                                          ;
                                     65  DD 00146      PUSHL   DIF$GL_MASFDB                                           ;
            00000000V         EF     02  FB 00148      CALLS   #2, COMPARE                                             ;
                              44     50  E8 0014F      BLBS    R0, 18$                                                 ;
                    0C        A5     65  D0 00152      MOVL    DIF$GL_MASFDB, DIF$GL_MASFDB+12                         ;
                    0C        A6     66  D0 00156      MOVL    DIF$GL_REVFDB, DIF$GL_REVFDB+12                         ;
                    10        A5     65  D0 0015A      MOVL    DIF$GL_MASFDB, DIF$GL_MASFDB+16                         ;
                    10        A6     66  D0 0015E      MOVL    DIF$GL_REVFDB, DIF$GL_REVFDB+16                         ;
                                00000000G  00  DD 00162      PUSHL   DIF$GL_WNDWSIZ                                  ;
            00000000V         EF         01  FB 00168      CALLS   #1, MISMATCH                                    ;
                              53         50  D0 0016F      MOVL    R0, STATUS                                      ;
            00000000G         3F         53  D1 00172      CMPL    STATUS, #DIF$_MAXDIF                            ;
                                         09  13 00179      BEQL    16$                                             ;
            00000000G         8F         53  D1 0017B      CMPL    STATUS, #DIF$_INSVIRMEM                         ;
                                         0A  12 00182      BNEQ    17$                                             ;
                    24        A5         08  8A 00184 16$:  BICB2   #8, DIF$GL_MASFDB+36                            ;
                    24        A6         08  8A 00188      BICB2   #8, DIF$GL_REVFDB+36                            ;
                                         43  11 0018C      BRB     23$                                             ;
                              52         65  D0 0018E 17$:  MOVL    DIF$GL_MASFDB, MASRDB                           ;
                              54         66  D0 00191      MOVL    DIF$GL_REVFDB, REVRDB                           ;
                                         33  11 00194      BRB     22$                                             ;
21                            67         05  E0 00196 18$:  BBS     #5, DIF$GL_FLAGS, 21$                           ;
1D                            67         06  E0 0019A      BBS     #6, DIF$GL_FLAGS, 21$                           ;
                                         67  95 0019E      TSTB    DIF$GL_FLAGS                                    ;
                                         19  19 001A0      BLSS    21$                                             ;
                              06   24     A5  E9 001A2      BLBC    DIF$GL_MASFDB+36, 19$                           ;
                                         02  DD 001A6      PUSHL   #2                                              ;
                                         55  DD 001A8      PUSHL   R5                                              ;
                                         08  11 001AA      BRB     20$                                             ;
```

```
                    0B      24   A6  E9 001AC 19$:    BLBC    DIF$GL_REVFDB+36, 21$                          ; 0603
                                 02  DD 001B0         PUSHL   #2                                             ; 0604
                                 56  DD 001B2         PUSHL   R6
    00000000G   00               02  FB 001B4 20$:    CALLS   #2, PUT_RECORD
                52               65  D0 001BB 21$:    MOVL    DIF$GL_MASFDB, MASRDB                          ; 0605
                54               66  D0 001BE         MOVL    DIF$GL_REVFDB, REVRDB                          ; 0606
            08  A4               10  88 001C1         BISB2   #16, 8(REVRDB)                                 ; 0607
            08  A2               10  88 001C5         BISB2   #16, 8(MASRDB)
03          08  A2               02  E0 001C9 22$:    BBS     #2, 8(MASRDB), 23$                             ; 0611
                             FF39    31 001CE         BRW     13$
            0C  A5           04  A5  D0 001D1 23$:    MOVL    DIF$GL_MASFDB+4, DIF$GL_MASFDB+12              ; 0616
            0C  A6           04  A6  D0 001D6         MOVL    DIF$GL_REVFDB+4, DIF$GL_REVFDB+12              ; 0617
            14  A5  00000000G   00  9E 001DB         MOVAB   DIF$GL_MASEOF, DIF$GL_MASFDB+20                ; 0618
            14  A6  00000000G   00  9E 001E3         MOVAB   DIF$GL_REVEOF, DIF$GL_REVFDB+20                ; 0619
                            1C  A5  D4 001EB         CLRL    DIF$GL_MASFDB+28                               ; 0620
                            1C  A6  D4 001EE         CLRL    DIF$GL_REVFDB+28                               ; 0621
    00000000G   00               00  FB 001F1         CALLS   #0, ADDITIONAL_OUTPUT                          ; 0622
    00000000G   8F               53  D1 001F8         CMPL    STATUS, #DIF$_MAXDIF                           ; 0627
                                 21  13 001FF         BEQL    25$
    00000000G   8F               53  D1 00201         CMPL    STATUS, #DIF$_INSVIRMEM
                                 18  13 00208         BEQL    25$
                    00000000G    00  D5 0020A         TSTL    DIF$GL_DIFSEC                                  ; 0628
                                 09  12 00210         BNEQ    24$
                53  00000000G    8F  D0 00212         MOVL    #DIF$_SAMEFILE, STATUS                         ; 0629
                                 07  11 00219         BRB     25$
                53  00000000G    8F  D0 0021B 24$:    MOVL    #DIF$_FILAREDIF, STATUS                        ; 0630
    00000000G   00               00  FB 00222 25$:    CALLS   #0, OUTPUT_LISTING_TRAILER                     ; 0635
                                 55  DD 00229         PUSHL   R5                                             ; 0640
    00000000G   00               01  FB 0022B         CALLS   #1, DIF$CLOSE_IN
                                 56  DD 00232         PUSHL   R6                                             ; 0641
    00000000G   00               01  FB 00234         CALLS   #1, DIF$CLOSE_IN
    00000000G   00               00  FB 0023B         CALLS   #0, DIF$CLOSE_OUT                              ; 0642
                50               53  D0 00242         MOVL    STATUS, R0                                     ; 0644
                                     04 00245         RET                                                    ; 0645

; Routine Size:  582 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  346     0646  1 ROUTINE mismatch (window) =
;  347     0647  2 BEGIN
;  348     0648  2
;  349     0649  2 !++
;  350     0650  2 !
;  351     0651  2 ! FUNCTIONAL DESCRIPTION:
;  352     0652  2 !
;  353     0653  2 !       This routine is called when a mismatch has been detected
;  354     0654  2 !       in the main loop.  For each file, we get a new record, and
;  355     0655  2 !       then compare it with all the records that we have on hand
;  356     0656  2 !       from the other file.  If a match is detected, the appropriate
;  357     0657  2 !       number of trailing records are examined to guarantee that it is
;  358     0658  2 !       a real match.  If no match is detected, the files are switched
;  359     0659  2 !       and the procedure is repeated.  If a match is not found within
;  360     0660  2 !       the specified WINDOW number of records from each file,
;  361     0661  2 !       then each file's COMP1REC pointer is moved forward one record
;  362     0662  2 !       and mismatch calls itself again.  Eventually, a match will be
;  363     0663  2 !       found, since every file is terminated by an EOF RDB that points
;  364     0664  2 !       to itself.
;  365     0665  2 !
;  366     0666  2 ! INPUTS:
;  367     0667  2 !
;  368     0668  2 !       window =  The size of the window to search for a match.
;  369     0669  2 !
;  370     0670  2 ! IMPLICIT INPUTS:
;  371     0671  2 !
;  372     0672  2 !       The COMP1REC pointers from the two FDB's mark the beginning
;  373     0673  2 !       of the comparisons.
;  374     0674  2 !
;  375     0675  2 ! OUTPUTS:
;  376     0676  2 !
;  377     0677  2 !       The file FDB's are updated so that they point to the next match.
;  378     0678  2 !
;  379     0679  2 ! ROUTINE VALUES:
;  380     0680  2 !
;  381     0681  2 !       Always true.
;  382     0682  2 !
;  383     0683  2 !--
;  384     0684  2
;  385     0685  2 LOCAL
;  386     0686  2     end_of_list,                    ! Flag indicating time to get a new record
;  387     0687  2     fdb1 : REF BBLOCK,              ! Local storage for input FDB addresses
;  388     0688  2     fdb2 : REF BBLOCK,
;  389     0689  2     fdb2_lastrec : REF BBLOCK,      ! Last record in list - set EOL flag
;  390     0690  2     tempfdb : REF BBLOCK,           ! Temp used to swap FDB addresses
;  391     0691  2     status;
;  392     0692  2
;  393     0693  2 fdb1 = dif$gl_masfdb;                               ! Init pointers to two FDB's
;  394     0694  2 fdb2 = dif$gl_revfdb;
;  395     0695  2
;  396     0696  3 INCRU i FROM 1 TO 2*(.window)                       ! Limit depth of search for next match
;  397     0697  3 DO BEGIN
;  398     0698  3
;  399     0699  3     IF (.dif$gl_difrec + (.i-1)/2 + 1) EQL .dif$gl_maxdif! If too many differnce records
;  400     0700  3        THEN RETURN print_and_quit (.i, dif$_maxdif);    ! Then tidy up and quit
;  401     0701  3
;  402     0702  4     IF NOT (status = get_record (.fdb1))            ! Get next record
```

```
;  403          0703  3      THEN RETURN print_and_quit (.i, .status);          ! If insvirmem, then get out
;  404          0704  3
;  405          0705  3  fdb1 [fdb$l_compnrec] = .fdb1 [fdb$l_currec];          ! Use record just fetched from one file
;  406          0706  3  fdb2 [fdb$l_compnrec] = .fdb2 [fdb$l_comprec];         ! Use first compare record from other file
;  407          0707  3  fdb2_lastrec = .fdb2 [fdb$l_currec];                   ! End with last record read from other file
;  408          0708  3  fdb2 [fdb$l_currec] = .fdb2 [fdb$l_comprec];          ! Init for test_match
;  409          0709  3
;  410          0710  3  end_of_list = false;                                   ! Init end of list flag
;  411          0711  3
;  412          0712  3  WHILE NOT .end_of_list                                 ! While not past end of list
;  413          0713  4  DO BEGIN                                              ! Compare against each record in other file's list
;  414          0714  4
;  415          0715  4     IF compare (.fdb1 [fdb$l_compnrec], .fdb2 [fdb$l_compnrec])     ! Compare two records
;  416          0716  5        THEN IF (status = test_match ())                 ! Same, then do enough records match
;  417          0717  5                THEN BEGIN                               ! Yes,
;  418          0718  5                     mark_match (dif$gl_masfdb);         ! Mark matched records
;  419          0719  5                     mark_match (dif$gl_revfdb);
;  420          0720  5                     write_mismatch();                   ! Output unmatched records
;  421          0721  5                     fdb1 [fdb$l_currec] = .fdb1 [fdb$l_compnrec];   ! Set CURREC's to first matches
;  422          0722  5                     fdb2 [fdb$l_currec] = .fdb2 [fdb$l_compnrec];
;  423          0723  5                     dif$gl_difrec = .dif$gl_difrec + (.i-1)/2 + 1;  ! Update difference count
;  424          0724  5                     IF .dif$gl_difrec EQL .dif$gl_maxdif    ! If too many differnce records
;  425          0725  5                        THEN RETURN (dif$_maxdif);           ! Then return the error
;  426          0726  5                     RETURN true;                        ! Return to dif$start
;  427          0727  5                     END
;  428          0728  5                                                         ! If not enough records in match
;  429          0729  4                ELSE IF .status EQL dif$_insvirmem       ! Then make sure we didn't run out o
;  430          0730  4                        THEN RETURN print_and_quit (.i, .status);   ! If we did, then quit
;  431          0731  4
;  432          0732  5        IF (.fdb2 [fdb$l_compnrec] EQL .fdb2_lastrec)    ! If at end of list
;  433          0733  4           THEN end_of_list = true                       ! Then set flag
;  434          0734  5           ELSE BEGIN                                    ! Else get next record
;  435          0735  5                fdb2 [fdb$l_compnrec] = ..fdb2 [fdb$l_currec];
;  436          0736  5                fdb2 [fdb$l_currec] = ..fdb2 [fdb$l_currec];
;  437          0737  4                END;
;  438          0738  4
;  439          0739  3        END;
;  440          0740  3
;  441          0741  3     tempfdb = .fdb1;                                    ! Exchange FDB's
;  442          0742  3     fdb1 = .fdb2;
;  443          0743  3     fdb2 = .tempfdb;
;  444          0744  3
;  445          0745  2     END;
;  446          0746  2
;  447          0747  2  fdb1 [fdb$l_comprec] = ..fdb1 [fdb$l_comprec];         ! Reset ptrs to first compare records
;  448          0748  2  fdb2 [fdb$l_comprec] = ..fdb2 [fdb$l_comprec];
;  449          0749  2
;  450          0750  2  dif$gl_difrec = .dif$gl_difrec + .window;              ! Update count of difference records
;  451          0751  2
;  452          0752  2  RETURN mismatch (1);                                   ! Recursively call mismatch with window size of 1
;  453          0753  1  END;                                                   ! Of mismatch
```

```
                                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11            ; 0646       ; Ro
                5B 00000000G  00  9E 00002                 MOVAB   DIF$GL_DIFREC, R11                              ;
                54 00000000G  00  9E 00009                 MOVAB   DIF$GL_MASFDB, FDB1                             ; 0693
                53 00000000G  00  9E 00010                 MOVAB   DIF$GL_REVFDB, FDB2                             ; 0694
58          04  AC            01  78 00017                 ASHL    #1, WINDOW, R8                                  ; 0696
                52            01  D0 0001C                 MOVL    #1, I                                           ;
                            00E4  31 0001F                 BRW     14$                                             ;
                50            6B  D0 00022 1$:             MOVL    DIF$GL_DIFREC, R0                               ; 0699
                55      FF    A2  9E 00025                 MOVAB   -1(R2), R5                                      ;
                55            02  C6 00029                 DIVL2   #2, R5                                          ;
                50      01  A540  9E 0002C                 MOVAB   1(R5)[R0], R0                                   ;
      00000000G 00            50  D1 00031                 CMPL    R0, DIF$GL_MAXDIF                               ;
                09            12  00038                    BNEQ    2$                                              ;
                   00000000G  8F  DD 0003A                 PUSHL   #DIF$_MAXDIF                                    ; 0700
                            0098  31 00040                 BRW     9$                                              ;
                              54  DD 00043 2$:             PUSHL   FDB1                                            ; 0702
      00000000V EF            01  FB 00045                 CALLS   #1, GET_RECORD                                  ;
                56            50  D0 0004C                 MOVL    R0, STATUS                                      ;
                03            56  E8 0004F                 BLBS    STATUS, 3$                                      ;
                            0084  31 00052                 BRW     8$                                              ;
            14  A4            64  D0 00055 3$:             MOVL    (FDB1), 20(FDB1)                                ; 0705
            14  A3      10    A3  D0 00059                 MOVL    16(FDB2), 20(FDB2)                              ; 0706
                5A            63  D0 0005E                 MOVL    (FDB2), FDB2_LASTREC                            ; 0707
                63      10    A3  D0 00061                 MOVL    16(FDB2), (FDB2)                                ; 0708
                              57  D4 00065                 CLRL    END_OF_LIST                                     ; 0710
                03            57  E9 00067 4$:             BLBC    END_OF_LIST, 5$                                 ; 0712
                            008E  31 0006A                 BRW     13$                                             ;
                        14    A3  DD 0006D 5$:             PUSHL   20(FDB2)                                        ; 0715
                        14    A4  DD 00070                 PUSHL   20(FDB1)                                        ;
      00000000V EF            02  FB 00073                 CALLS   #2, COMPARE                                     ;
                68            50  E9 0007A                 BLBC    R0, 10$                                         ;
      00000000V EF            00  FB 0007D                 CALLS   #0, TEST_MATCH                                  ; 0716
                56            50  D0 00084                 MOVL    R0, STATUS                                      ;
                46            56  E9 00087                 BLBC    STATUS, 7$                                      ;
                   00000000G  00  9F 0008A                 PUSHAB  DIF$GL_MASFDB                                   ; 0718
      00000000V EF            01  FB 00090                 CALLS   #1, MARK_MATCH                                  ;
                   00000000G  00  9F 00097                 PUSHAB  DIF$GL_REVFDB                                   ; 0719
      00000000V EF            01  FB 0009D                 CALLS   #1, MARK_MATCH                                  ;
      00000000G 00            00  FB 000A4                 CALLS   #0, WRITE_MISMATCH                              ; 0720
                64      14    A4  D0 000AB                 MOVL    20(FDB1), (FDB1)                                ; 0721
                63      14    A3  D0 000AF                 MOVL    20(FDB2), (FDB2)                                ; 0722
                50            6B  D0 000B3                 MOVL    DIF$GL_DIFREC, R0                               ; 0723
                6B      01  A540  9E 000B6                 MOVAB   1(R5)[R0], DIF$GL_DIFREC                        ;
      00000000G 00            6B  D1 000BB                 CMPL    DIF$GL_DIFREC, DIF$GL_MAXDIF                    ; 0724
                08            12  000C2                    BNEQ    6$                                              ;
                50 00000000G  8F  D0 000C4                 MOVL    #DIF$_MAXDIF, R0                                ; 0725
                              04  000CB                    RET                                                     ;
                50            01  D0 000CC 6$:             MOVL    #1, R0                                          ; 0726
                              04  000CF                    RET                                                     ;
      00000000G 8F            56  D1 000D0 7$:             CMPL    STATUS, #DIF$_INSVIRMEM                         ; 0729
                0C            12  000D7                    BNEQ    10$                                             ;
                              56  DD 000D9 8$:             PUSHL   STATUS                                          ; 0730
                              52  DD 000DB 9$:             PUSHL   I                                               ;
      00000000V EF            02  FB 000DD                 CALLS   #2, PRINT_AND_QUIT                              ;
                              04  000E4                    RET                                                     ;
                5A      14    A3  D1 000E5 10$:            CMPL    20(FDB2), FDB2_LASTREC                          ; 0732
                05            12  000E9                    BNEQ    11$                                             ;
```

```
                    57              01  DO 000EB          MOVL    #1, END_OF_LIST                         ; 0733
                                    08  11 000EE          BRB     12$                                     ;
                14  A3      00      B3  DO 000F0 11$:     MOVL    @0(FDB2), 20(FDB2)                      ; 0735
                    73              93  DO 000F5          MOVL    @(FDB2)+, -(FDB2)                       ; 0736
                                FF6C    31 000F8 12$:     BRW     4$                                      ; 0712
                    59              54  DO 000FB 13$:     MOVL    FDB1, TEMPFDB                           ; 0741
                    54              53  DO 000FE          MOVL    FDB2, FDB1                              ; 0742
                    53              59  DO 00101          MOVL    TEMPFDB, FDB2                           ; 0743
                                    52  D6 00104          INCL    I                                       ; 0696
                    58              52  D1 00106 14$:     CMPL    I, R8                                   ;
                                    03  1A 00109          BGTRU   15$                                     ;
                                FF14    31 0010B          BRW     1$                                      ;
                10  A4      10      B4  DO 0010E 15$:     MOVL    @16(FDB1), 16(FDB1)                     ; 0747
                10  A3      10      B3  DO 00113          MOVL    @16(FDB2), 16(FDB2)                     ; 0748
                    6B      04      AC  C0 00118          ADDL2   WINDOW, DIF$GL_DIFREC                   ; 0750
                                    01  DD 0011C          PUSHL   #1                                      ; 0752
            FEDD    CF              01  FB 0011E          CALLS   #1, MISMATCH                            ;
                                        04 00123          RET                                             ; 0753

; Routine Size:  292 bytes,    Routine Base:  $CODE$ + 0246
```

```
;  455     0754  1 ROUTINE test_match =
;  456     0755  2 BEGIN
;  457     0756  2
;  458     0757  2 !++
;  459     0758  2 !
;  460     0759  2 ! FUNCTIONAL DESCRIPTION:
;  461     0760  2 !
;  462     0761  2 !       This routine is called when a match has been detected
;  463     0762  2 !       by mismatch.  It checks to see that the match is long
;  464     0763  2 !       enough to be a legal match, and signals its result via
;  465     0764  2 !       the return status.
;  466     0765  2 !
;  467     0766  2 ! INPUTS:
;  468     0767  2 !
;  469     0768  2 !       None.
;  470     0769  2 !
;  471     0770  2 ! OUTPUTS:
;  472     0771  2 !
;  473     0772  2 !       None.
;  474     0773  2 !
;  475     0774  2 ! ROUTINE VALUES:
;  476     0775  2 !
;  477     0776  2 !       True, if the match is long enough.
;  478     0777  2 !       False, if it is not long enough.
;  479     0778  2 !       DIF$C_INSVIRMEM, if get_record fails due to insufficient VM.
;  480     0779  2 !
;  481     0780  2 !--
;  482     0781  2 LOCAL
;  483     0782  2     status;
;  484     0783  2
;  485     0784  2
;  486     0785  2 INCR i FROM 1 TO .dif$gl_match - 1                      ! Do DIF$GL_MATCH - 1 number of compares
;  487     0786  3 DO BEGIN
;  488     0787  3
;  489     0788  4     IF NOT (status = get_record (dif$gl_masfdb))        ! Get the next record from each file
;  490     0789  3         THEN RETURN .status;
;  491     0790  4     IF NOT (status = get_record (dif$gl_revfdb))
;  492     0791  3         THEN RETURN .status;
;  493     0792  3
;  494     0793  4     IF NOT (compare (.dif$gl_masfdb [fdb$l_currec],     ! Compare the two fetched records
;  495     0794  4                      .dif$gl_revfdb [fdb$l_currec]))
;  496     0795  4         THEN BEGIN                                      ! If different, then restore ptrs
;  497     0796  4              dif$gl_masfdb [fdb$l_currec] = .dif$gl_masfdb [fdb$l_compnrec];
;  498     0797  4              dif$gl_revfdb [fdb$l_currec] = .dif$gl_revfdb [fdb$l_compnrec];
;  499     0798  4              RETURN false;                              ! Return false
;  500     0799  3              END;
;  501     0800  3
;  502     0801  2     END;
;  503     0802  2
;  504     0803  2 RETURN true;                                            ! Same, return true
;  505     0804  1 END;                                                    ! Of test_match


                        00FF 00000 TEST_MATCH:
```

```
                                                 .WORD   Save R2,R3,R4,R5,R6,R7                  ; 0754
                57 00000000V   EF   9E 00002     MOVAB   GET_RECORD, R7                          ;
                56 00000000G   00   9E 00009     MOVAB   DIF$GL_REVFDB, R6                       ;
                55 00000000G   00   9E 00010     MOVAB   DIF$GL_MASFDB, R5                       ;
                54 00000000G   00   D0 00017     MOVL    DIF$GL_MATCH, R4                        ; 0785
                               53   D4 0001E     CLRL    I                                       ;
                               32   11 00020     BRB     4$                                      ;
                               55   DD 00022 1$:  PUSHL   R5                                      ; 0788
                67                  01   FB 00024     CALLS   #1, GET_RECORD                      ;
                52                  50   D0 00027     MOVL    R0, STATUS                         ;
                0B                  52   E9 0002A     BLBC    STATUS, 2$                         ;
                                    56   DD 0002D     PUSHL   R6                                 ; 0790
                67                  01   FB 0002F     CALLS   #1, GET_RECORD                     ;
                52                  50   D0 00032     MOVL    R0, STATUS                         ;
                04                  52   E8 00035     BLBS    STATUS, 3$                         ;
                50                  52   D0 00038 2$:  MOVL    STATUS, R0                         ; 0791
                                    04   0003B        RET                                        ;
                                    66   DD 0003C 3$:  PUSHL   DIF$GL_REVFDB                     ; 0794
                                    65   DD 0003E     PUSHL   DIF$GL_MASFDB                      ; 0793
          00000000V EF              02   FB 00040     CALLS   #2, COMPARE                        ;
                0A                  50   E8 00047     BLBS    R0, 4$                             ;
                65        14        A5   D0 0004A     MOVL    DIF$GL_MASFDB+20, DIF$GL_MASFDB    ; 0796
                66        14        A6   D0 0004E     MOVL    DIF$GL_REVFDB+20, DIF$GL_REVFDB    ; 0797
                                    08   11 00052     BRB     5$                                 ; 0798
             CA 53                  54   F2 00054 4$:  AOBLSS  R4, I, 1$                          ; 0785
                50                  01   D0 00058     MOVL    #1, R0                             ; 0803
                                    04   0005B        RET                                        ;
                                    50   D4 0005C 5$:  CLRL    R0                                 ; 0804
                                    04   0005E        RET                                        ;

; Routine Size:  95 bytes,    Routine Base:  $CODE$ + 036A
```

```
;  507          0805  1 ROUTINE compare (rdb1, rdb2) =
;  508          0806  2 BEGIN
;  509          0807  2
;  510          0808  2 !++
;  511          0809  2 !
;  512          0810  2 ! FUNCTIONAL DESCRIPTION:
;  513          0811  2 !
;  514          0812  2 !       This routine is to compare to records.  Since every
;  515          0813  2 !       record has already been preprocessed by the time it gets
;  516          0814  2 !       here, all this routine needs to do is check the length
;  517          0815  2 !       and content of the records.
;  518          0816  2 !
;  519          0817  2 ! INPUTS:
;  520          0818  2 !
;  521          0819  2 !       rdb1, rdb2 =    The addresses of the RDB's of the two records
;  522          0820  2 !                       that are to be compared.
;  523          0821  2 !
;  524          0822  2 ! OUTPUTS:
;  525          0823  2 !
;  526          0824  2 !       None.
;  527          0825  2 !
;  528          0826  2 ! ROUTINE VALUES:
;  529          0827  2 !
;  530          0828  2 !       True, if the records are the same.
;  531          0829  2 !       False, if the records differ.
;  532          0830  2 !
;  533          0831  2 !--
;  534          0832  2
;  535          0833  2 MAP
;  536          0834  2     rdb1 : REF BBLOCK,
;  537          0835  2     rdb2 : REF BBLOCK;
;  538          0836  2
;  539          0837  2 IF .rdb1 [rdb$w_length] NEQ .rdb2 [rdb$w_length]        ! Compare lengths
;  540          0838  2    THEN RETURN false;
;  541          0839  2
;  542          0840  2 IF .rdb1 [rdb$v_eof] AND .rdb2 [rdb$v_eof]              ! Special case EOF's
;  543          0841  2    THEN RETURN true;
;  544          0842  2
;  545          0843  2 IF CH$NEQ (.rdb1 [rdb$w_length], rdb1 [rdb$t_text],    ! Compare content
;  546          0844  2            .rdb2 [rdb$w_length], rdb2 [rdb$t_text])
;  547          0845  2    THEN RETURN false
;  548          0846  2    ELSE RETURN true;
;  549          0847  2
;  550          0848  1 END;

                                                  000C 00000 COMPARE:.WORD   Save R2,R3                                  ; 0805
                                  51        04    AC   D0 00002          MOVL   RDB1, R1                                 ; 0837
                                  50        08    AC   D0 00006          MOVL   RDB2, R0                                 ;
                            12    A0        12    A1   B1 0000A          CMPW   18(R1), 18(R0)                           ;
                                                  1A   12 0000F          BNEQ   3$                                       ;
                      05          08        A1         02   E1 00011     BBC    #2, 8(R1), 1$                            ; 0840
                      0C          08        A0         02   E0 00016     BBS    #2, 8(R0), 2$                            ;
                12    A0          00        14    A1   12   A1 2D 0001B 1$:  CMPC5  18(R1), 20(R1), #0, 18(R0), 20(R0)  ; 0844
```

```
                     14  A0      00023
                         04  12  00025          BNEQ    3$
                     50      01  D0  00027 2$:   MOVL    #1, R0                                  0846
                             04  0002A          RET
                     50      04  D4  0002B 3$:   CLRL    R0                                      0848
                             04  0002D          RET

; Routine Size:  46 bytes,    Routine Base:  $CODE$ + 03C9
```

```
;  552          0849  1 ROUTINE mark_match (fdb) =                                                                    ; R
;  553          0850  2 BEGIN
;  554          0851  2
;  555          0852  2 !++
;  556          0853  2 !
;  557          0854  2 ! FUNCTIONAL DESCRIPTION:
;  558          0855  2 !
;  559          0856  2 !       This routine is called to mark the most recent set of matched records.
;  560          0857  2 !       The match records start with COMPNREC and extend DIF$GL_MATCH number of
;  561          0858  2 !       records from there.
;  562          0859  2 !
;  563          0860  2 ! INPUTS:
;  564          0861  2 !
;  565          0862  2 !       fdb =   The address of the FDB for the file whose matched records are
;  566          0863  2 !               to be marked.
;  567          0864  2 !
;  568          0865  2 ! OUTPUTS:
;  569          0866  2 !
;  570          0867  2 !       The MATCH bit in the RDB's of the matched records are set.
;  571          0868  2 !       The MATCHONE bit in the RDB of the first match record is set.
;  572          0869  2 !
;  573          0870  2 ! ROUTINE VALUES:
;  574          0871  2 !
;  575          0872  2 !       Always true.
;  576          0873  2 !
;  577          0874  2 !--
;  578          0875  2
;  579          0876  2 MAP
;  580          0877  2     fdb : REF BBLOCK;
;  581          0878  2
;  582          0879  2 LOCAL
;  583          0880  2     rdb : REF BBLOCK;                   ! Address of current RDB
;  584          0881  2
;  585          0882  2 rdb = .fdb [fdb$l_compnrec];            ! Locate first match
;  586          0883  2 rdb [rdb$v_matchone] = true;            ! Mark it as first match
;  587          0884  2
;  588          0885  2 INCR i FROM 1 TO .dif$gl_match          ! Locate all matches
;  589          0886  3 DO BEGIN
;  590          0887  3    rdb [rdb$v_match]   true;           ! Mark each as a match
;  591          0888  3    rdb = .rdb [rdb$l_flink];           ! Get next match
;  592          0889  2    END;
;  593          0890  2
;  594          0891  2 RETURN true;
;  595          0892  1 END;


                                0000 00000 MARK_MATCH:
                                           .WORD   Save nothing                                                     ; 0849
                     50    04   AC  D0 00002        MOVL    FDB, R0                                                 ; 0882
                     50    14   A0  D0 00006        MOVL    20(R0), RDB                                             ;
                08   A0         20  88 0000A        BISB2   #32, 8(RDB)                                             ; 0883
                                51  D4 0000E        CLRL    I                                                       ; 0885
                                07  11 00010        BRB     2$                                                      ;
                08   A0         10  88 00012 1$:    BISB2   #16, 8(RDB)                                             ; 0887
```

```
                    50              60  D0 00016          MOVL    (RDB), RDB                                  ; 0888
                F1  51 00000000G    00  F3 00019 2$:      AOBLEQ  DIF$GL_MATCH, I, 1$                         ; 0885
                    50              01  D0 00021          MOVL    #1, R0                                      ; 0891
                                        04 00024          RET                                                 ; 0892

; Routine Size:  37 bytes,    Routine Base:  $CODE$ + 03F7
```

```
;  597          0893  1 %ROUTINE get_record (fdb) =
;  598          0894  2 BEGIN
;  599          0895  2                                                                                ; R
;  600          0896  2 !++
;  601          0897  2 !
;  602          0898  2 ! FUNCTIONAL DESCRIPTION:
;  603          0899  2 !
;  604          0900  2 !       This routine is called to supply the next record, from the specified
;  605          0901  2 !       file.  It determines whether there is a need to go to the file for
;  606          0902  2 !       the record (i.e., it checks to see if an RDB already exists for the
;  607          0903  2 !       record), and it keeps examining records until it finds one that is
;  608          0904  2 !       not supposed to be ignored in all comparisons.
;  609          0905  2 !
;  610          0906  2 ! INPUTS:
;  611          0907  2 !
;  612          0908  2 !       fdb =   The address of the FDB for the input file.  CURREC specifies
;  613          0909  2 !               the record just before the one to be fetched.
;  614          0910  2 !
;  615          0911  2 ! OUTPUTS:
;  616          0912  2 !
;  617          0913  2 !       The address of the fetched record is strored in the CURREC
;  618          0914  2 !       field of the FDB.
;  619          0915  2 !
;  620          0916  2 ! ROUTINE VALUES:
;  621          0917  2 !
;  622          0918  2 !       DIF$_INSVIRMEM, if read_record failed due to insufficient VM,
;  623          0919  2 !       true, otherwise.
;  624          0920  2 !
;  625          0921  2 !--
;  626          0922  2
;  627          0923  2 MAP
;  628          0924  2     fdb : REF BBLOCK;
;  629          0925  2
;  630          0926  2 LOCAL
;  631          0927  2     rdb : REF BBLOCK,                                  ! Address of current RDB
;  632          0928  2     status;
;  633          0929  2
;  634          0930  2 status = true;                                         ! Assume no problem reading record
;  635          0931  2
;  636          0932  3 DO BEGIN
;  637          0933  3    IF (rdb = .fdb [fdb$l_currec]) EQL 0                ! If no previous record in memory
;  638          0934  3         THEN status = read_record (.fdb)               ! then read a new record from the file
;  639          0935  3         ELSE IF (rdb = .rdb [rdb$l_flink]) EQL 0       ! or, if next record is not in memory
;  640          0936  3                 THEN status = read_record (.fdb);      ! then read a new record
;  641          0937  3    IF NOT .status                                      ! If error reading
;  642          0938  3       THEN RETURN .status;                             ! Then return error
;  643          0939  3    IF .rdb EQL 0                                       ! If new record read
;  644          0940  3       THEN rdb = .fdb [fdb$l_lastrec];                 ! then get its address
;  645          0941  3    fdb [fdb$l_currec] = .rdb;                          ! Set current field to fetched record
;  646          0942  3    END
;  647          0943  3
;  648          0944  2 UNTIL (NOT .rdb [rdb$v_ignored]);                      ! Read records until not ignored
;  649          0945  2
;  650          0946  2 RETURN true;
;  651          0947  1 END;                                                   ! Of get_record
```

```
                                 000C 00000 GET_RECORD:
                                                .WORD   Save R2,R3                      ; 0893
                         50              01  D0 00002          MOVL    #1, STATUS          ; 0930
                         53      04      AC  D0 00005          MOVL    FDB, R3             ; 0933
                         52              63  D0 00009 1$:      MOVL    (R3), RDB           ;
                                         05  13 0000C          BEQL    2$                  ;
                         52              62  D0 0000E          MOVL    (RDB), RDB          ; 0935
                                         09  12 00011          BNEQ    3$                  ;
                                         53  DD 00013 2$:      PUSHL   R3                  ; 0936
                 00000000V               EF  FB 00015          CALLS   #1, READ_RECORD     ;
                         12              50  E9 0001C 3$:      BLBC    STATUS, 5$          ; 0937
                                         52  D5 0001F          TSTL    RDB                 ; 0939
                                         04  12 00021          BNEQ    4$                  ;
                         52      08      A3  D0 00023          MOVL    8(R3), RDB          ; 0940
                         63              52  D0 00027 4$:      MOVL    RDB, (R3)           ; 0941
                         DB      08      A2  E8 0002A          BLBS    8(RDB), 1$          ; 0944
                         50              01  D0 0002E          MOVL    #1, R0              ; 0946
                                             04 00031 5$:      RET                         ; 0947

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 041C
```

```
;  653     0948  1 ROUTINE read_record (fdb) =
;  654     0949  2 BEGIN
;  655     0950  2
;  656     0951  2 !++
;  657     0952  2 !
;  658     0953  2 ! FUNCTIONAL DESCRIPTION:
;  659     0954  2 !
;  660     0955  2 !       This routine is called to get the next record from the specified
;  661     0956  2 !       input file, put it in an RDB, process it for any ignore fields or
;  662     0957  2 !       characters, and link it's RDB with already existing RDB's.
;  663     0958  2 !
;  664     0959  2 ! INPUTS:
;  665     0960  2 !                                                                              ; R
;  666     0961  2 !       fdb =   The address of the FDB for the input file.
;  667     0962  2 !
;  668     0963  2 ! OUTPUTS:
;  669     0964  2 !
;  670     0965  2 !       The RDB is allocated, filled in, and returned in the LASTREC field of the FDB.
;  671     0966  2 !
;  672     C967  2 ! ROUTINE VALUES:
;  673     0968  2 !
;  674     0969  2 !       DIF$_INSVIRMEM, if RDB allocation fails due to insufficient VM,
;  675     0970  2 !       true otherwise.
;  676     0971  2 !
;  677     0972  2 !--
;  678     0973  2
;  679     0974  2 MAP
;  680     0975  2     fdb : REF BBLOCK;
;  681     0976  2
;  682     0977  2 LOCAL
;  683     0978  2     lastrdb : REF BBLOCK,               ! Local for last RDB in list
;  684     0979  2     rab : REF BBLOCK,                   ! Local for file RAB
;  685     0980  2     rdb : REF BBLOCK,                   ! Local for RDB just allocated
;  686     0981  2     status;
;  687     0982  2
;  688     0983  2 !
;  689     0984  2 ! Try to get a record from the file.  Handle special EOF case.
;  690     0985  2 !
;  691     0986  2 rab = .fdb [fdb$l_rabpt;];                                      ! Get address of RAB
;  692     0987  3 IF NOT (status = $GET (RAB = .rab))                             ! Get record from file
;  693     0988  2    THEN IF .status NEQ RMS$_EOF                                 ! If error and not EOF
;  694     0989  2            THEN SIGNAL_STOP (dif$_readerr,                      ! Then signal error
;  695     0990  2                              1, .fdb [fdb$l_fildesc], .status,
;  696     0991  2                              .rab [rab$l_stv])
;  697     0992  3            ELSE BEGIN                                           ! Else link static EOF RDB to list
;  698     0993  3
;  699     0994  3                 IF (rdb = .fdb [fdb$l_lastrec]) EQL 0           ! If first record
;  700     0995  4                    THEN BEGIN                                   ! Then use as head of list
;  701     0996  4                         fdb [fdb$l_firstrec] = .fdb [fdb$l_eofrec];
;  702     0997  4                         fdb [fdb$l_comprec] = .fdb [fdb$l_eofrec];
;  703     0998  4                         END
;  704     0999  3                    ELSE rdb [rdb$l_flink] = .fdb [fdb$l_eofrec];        ! Else link to end of list
;  705     1000  3
;  706     1001  3                 fdb [fdb$l_lastrfa] = .rdb;                     ! Remember last successful read
;  707     1002  3                 rdb = .fdb [fdb$l_eofrec];                      ! Get address of EOF RDB
;  708     1003  3                 fdb [fdb$l_numrec] = .fdb [fdb$l_numrec] + 1;   ! Incr record number in FDB
;  709     1004  3                 rdb [rdb$l_number] = .fdb [fdb$l_numrec];       ! Assign EOF record number in RDB
```

```
;  710          1005  3                fdb [fdb$l_lastrec] = .rdb;                        ! Update FDB last record read ptr
;  711          1006  3
;  712          1007  3                RETURN true;
;  713          1008  2                END;
;  714          1009  2
;  715          1010  2 !
;  716          1011  2 ! Record successfully read from file.  So get an RDB, link it to the list, and fill in some
;  717          1012  2 ! simple fields in the RDB and FDB.
;  718          1013  2 !
;  719          1014  3 IF NOT (status = allocate_rdb (rdb, .rab [rab$w_rsz]))          ! Allocate a RDB for the new record
;  720          1015  2    THEN RETURN .status;                                          ! Return if unsuccessful
;  721          1016  2
;  722          1017  2 IF (lastrdb = .fdb [fdb$l_lastrec]) EQL 0                       ! If first record
;  723          1018  3    THEN BEGIN                                                    ! Then use as head of list
;  724          1019  3         fdb [fdb$l_firstrec] = .rdb;
;  725          1020  3         fdb [fdb$l_complrec] = .rdb;
;  726          1021  3         END
;  727          1022  2    ELSE lastrdb [rdb$l_flink] = .rdb;                            ! Else link to end of list
;  728          1023  2
;  729          1024  2 fdb [fdb$l_lastrec] = .rdb;                                    ! Update FDB last record read ptr
;  730          1025  2
;  731          1026  2 fdb [fdb$l_numrec] = .fdb [fdb$l_numrec] + 1;                  ! Incr number of record in FDB
;  732          1027  2 rdb [rdb$l_number] = .fdb [fdb$l_numrec];                      ! Assign record number in RDB
;  733          1028  2 rdb [rdb$w_length] = .rab [rab$w_rsz];                        ! Move record size into the RDB
;  734          1029  2
;  735          1030  2 !
;  736          1031  2 ! If record size is non-zero, then move the text and RFA into the RDB and
;  737          1032  2 ! process the record for any ignore characters.  Otherwise, check if we are
;  738          1033  2 ! ignoring blank lines, and if we are, set ignore flag for this record.
;  739          1034  2 !
;  740          1035  2 CH$MOVE (rfa$c_size, rab [rab$w_rfa], rdb [rdb$w_rfa]);        ! Get RFA
;  741          1036  2 IF .rdb [rdb$w_length] GTRU 0                                  ! Is length non-zero?
;  742          1037  3    THEN BEGIN                                                    ! Yes
;  743          1038  3         CH$MOVE (.rdb [rdb$w_length],                         ! Get text
;  744          1039  3                  .rab [rab$l_rbf],
;  745          1040  3                  rdb [rdb$t_text]);
;  746          1041  3         IF (.dif$gl_ignore AND NOT (ign$m_exact OR ign$m_pretty)) NEQ 0      ! If some edit flag is set
;  747          1042  3            THEN process_record (.fdb);                        ! Process the text
;  748          1043  3         END
;  749          1044  3
;  750          1045  2    ELSE IF .dif$gl_ignore [ign$v_blnklin]                     ! No, do we ignore blank lines
;  751          1046  2            THEN rdb [rdb$v_ignored] = true;                   ! Yes, then mark the RDB
;  752          1047  2
;  753          1048  2 RETURN true;
;  754          1049  1 END;                                                          ! Of read_record


                                                .EXTRN  SYS$GET

                                03FC 00000 READ_RECORD:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 0948
                    59 00000000G   00  9E 00002         MOVAB   DIF$GL_IGNORE, R9                       ;
                    5E             04  C2 00009         SUBL2   #4, SP                                  ;
                    56       04    AC  D0 0000C         MOVL    FDB, R6                                 ; 0986
                    58       30    A6  D0 00010         MOVL    48(R6), RAB                             ;
                                   58  DD 00014         PUSHL   RAB                                     ; 0987
```

```
             0000000G  00           01  FB 00016        CALLS   #1, SYS$GET                              ;
                       50           50  D0 0001D        MOVL    R0, STATUS                               ;
                       52           52  E8 00020        BLBS    STATUS, 4$                               ;
             0001827A  8F           52  D1 00023        CMPL    STATUS, #98938                           ; 0988
                                    19  13 0002A        BEQL    1$                                       ;
                             0C     A8  DD 0002C        PUSHL   12(RAB)                                  ; 0991
                                    52  DD 0002F        PUSHL   STATUS                                   ; 0990
                             38     A6  DD 00031        PUSHL   56(R6)                                   ;
                                    01  DD 00034        PUSHL   #1                                       ; 0989
                      00000000G     8F  DD 00036        PUSHL   #DIF$_READERR                            ;
             00000000G  00          05  FB 0003C        CALLS   #5, LIB$STOP                             ;
                                    30  11 00043        BRB     4$                                       ;
                       6E    08     A6  D0 00045 1$:    MOVL    8(R6), RDB                               ; 0994
                                    0C  12 00049        BNEQ    2$                                       ;
                  04   A6    18     A6  D0 0004B        MOVL    24(R6), 4(R6)                            ; 0996
                  10   A6    18     A6  D0 00050        MOVL    24(R6), 16(R6)                           ; 0997
                                    05  11 00055        BRB     3$                                       ; 0994
                  00   BE    18     A6  D0 00057 2$:    MOVL    24(R6), @RDB                             ; 0999
                  1C   A6           6E  D0 0005C 3$:    MOVL    RDB, 28(R6)                              ; 1001
                       6E    18     A6  D0 00060        MOVL    24(R6), RDB                              ; 1002
                             20     A6  D6 00064        INCL    32(R6)                                   ; 1003
                       50           6E  D0 00067        MOVL    RDB, R0                                  ; 1004
                  04   A0    20     A6  D0 0006A        MOVL    32(R6), 4(R0)                            ;
                  08   A6           50  D0 0006F        MOVL    R0, 8(R6)                                ; 1005
                                    6C  11 00073        BRB     9$                                       ; 1007
                       7E    22     A8  3C 00075 4$:    MOVZWL  34(RAB), -(SP)                           ; 1014
                             04     AE  9F 00079        PUSHAB  RDB                                      ;
             00000000V EF           02  FB 0007C        CALLS   #2, ALLOCATE_RDB                         ;
                       52           50  D0 00083        MOVL    R0, STATUS                               ;
                       04           52  E8 00086        BLBS    STATUS, 5$                               ;
                       50           52  D0 00089        MOVL    STATUS, R0                               ; 1015
                                        04 0008C        RET                                              ;
                       57           6E  D0 0008D 5$:    MOVL    RDB, R7                                  ; 1019
                       50    08     A6  D0 00090        MOVL    8(R6), LASTRDB                           ; 1017
                                    0A  12 00094        BNEQ    6$                                       ;
                  04   A6           57  D0 00096        MOVL    R7, 4(R6)                                ; 1019
                  10   A6           57  D0 0009A        MOVL    R7, 16(R6)                               ; 1020
                                    03  11 0009E        BRB     7$                                       ; 1017
                       60           57  D0 000A0 6$:    MOVL    R7, (LASTRDB)                            ; 1022
                  08   A6           57  D0 000A3 7$:    MOVL    R7, 8(R6)                                ; 1024
                             20     A6  D6 000A7        INCL    32(R6)                                   ; 1026
                  04   A7    20     A6  D0 000AA        MOVL    32(R6), 4(R7)                            ; 1027
                  12   A7    22     A8  B0 000AF        MOVW    34(RAB), 18(R7)                          ; 1028
    0C     A7     10   A8           06  28 000B4        MOVC3   #6, 16(RAB), 12(R7)                      ; 1035
                             12     A7  B5 000BA        TSTW    18(R7)                                   ; 1036
                                    1B  13 000BD        BEQL    8$                                       ;
    14     A7     28   B8    12     A7  28 000BF        MOVC3   18(R7), @40(RAB), 20(R7)                 ; 1040
             FFFFFF3F  8F           69  D3 000C6        BITL    DIF$GL_IGNORE, #-193                     ; 1041
                                    12  13 000CD        BEQL    9$                                       ;
                                    56  DD 000CF        PUSHL   R6                                       ; 1042
             00000000V EF           01  FB 000D1        CALLS   #1, PROCESS_RECORD                       ;
                                    07  11 000D8        BRB     9$                                       ; 1036
                       04           69  E9 000DA 8$:    BLBC    DIF$GL_IGNORE, 9$                        ; 1045
                  08   A7           01  88 000DD        BISB2   #1, 8(R7)                                ; 1046
                       50           01  D0 000E1 9$:    MOVL    #1, R0                                   ; 1048
                                        04 000E4        RET                                              ; 1049
```

; Routine Size: 229 bytes, Routine Base: $CODE$ + 044E

; R

```
;  756          1050  1 ROUTINE process_record (fdb) =
;  757          1051  2 BEGIN
;  758          1052  2
;  759          1053  2 !++
;  760          1054  2 !
;  761          1055  2 ! FUNCTIONAL DESCRIPTION:
;  762          1056  2 !
;  763          1057  2 !       This routine is called to remove all ignore characters from a record that
;  764          1058  2 !       has just been read.
;  765          1059  2 !
;  766          1060  2 ! INPUTS:
;  767          1061  2 !
;  768          1062  2 !       fdb =       The address of the FDB for the input file.  LASTREC contains the
;  769          1063  2 !                   address of the RDB for the record that is to be processed.
;  770          1064  2 !
;  771          1065  2 ! OUTPUTS:
;  772          1066  2 !
;  773          1067  2 !       The LENGTH and TEXT fields of the RDB are modified to reflect any editing
;  774          1068  2 !       that was performed.  The IGNORE flag is set if all text has been deleted and
;  775          1069  2 !       we are ignoring blank lines.
;  776          1070  2 !
;  777          1071  2 ! ROUTINE VALUES:
;  778          1072  2 !
;  779          1073  2 !       Always true.
;  780          1074  2 !
;  781          1075  2 !--
;  782          1076  2
;  783          1077  2 MAP
;  784          1078  2     fdb : REF BBLOCK;
;  785          1079  2
;  786          1080  2 LOCAL
;  787          1081  2     rdb : REF BBLOCK,                          ! Address of RDB of record to process
;  788          1082  2     blankseen,                                 ! Flag indicating last char was a blank
;  789          1083  2     ignore,                                    ! Flag indicating this char should be ignored
;  790          1084  2     working_len,                               ! Local working length of record
;  791          1085  2     charptr : REF VECTOR [,BYTE],              ! Ptr to char being tested
;  792          1086  2     moveptr : REF VECTOR [,BYTE];              ! Ptr to byte after last moved char
;  793          1087  2
;  794          1088  2 rdb = .fdb [fdb$l_lastrec];                   ! Get RDB of record to process
;  795          1089  2 charptr = rdb [rdb$t_text];                   ! Point to start of text
;  796          1090  2 working_len = .rdb [rdb$w_length];            ! Working copy of the lnegth
;  797          1091  2
;  798          1092  2 !
;  799          1093  2 ! If ignoring headers, and if this record is part of the header, then ignore it.
;  800          1094  2 !
;  801          1095  2 IF .dif$gl_ignore [ign$v_header]               ! If ignoring headers
;  802          1096  2    THEN IF .fdb [fdb$l_headcnt] LSSU .dif$gl_header   ! And if record is part of header
;  803          1097  3         THEN BEGIN                            ! Then ignore it
;  804          1098  3              fdb [fdb$l_headcnt] =            ! Incr count of header records
;  805          1099  3                  .fdb [fdb$l_headcnt] + 1;    !
;  806          1100  3              rdb [rdb$v_ignored] = true;      ! Mark the record
;  807          1101  3              RETURN true;                     ! Return
;  808          1102  3              END                              !
;  809          1103  3
;  810          1104  3         ELSE IF (.working_len NEQ 0)          ! If first char is <FF>
;  811          1105  3                  AND (.charptr [0] EQL %X'0C')!
;  812          1106  3              THEN BEGIN                       !
```

```
;  813     1107  3                    IF .working_len EQL 1                       ! Then if only one char in record
;  814     1108  3                       THEN fdb [fdb$l_headcnt] = 0             ! Then start header with next record
;  815     1109  4                       ELSE BEGIN                               ! Then start a new header
;  816     1110  4                            fdb [fdb$l_headcnt] = 1;            ! One record found
;  817     1111  4                            rdb [rdb$v_ignored] = true;         ! Ignore the record
;  818     1112  3                            END;                                !
;  819     1113  3                    RETURN true;                                ! Return done
;  820     1114  2                    END;                                        !
;  821     1115  2
;  822     1116  2
;  823     1117  2 !
;  824     1118  2 ! If ignoring comments, and if part of this record is a comment,
;  825     1119  2 ! then ignore that part of the record.
;  826     1120  2 !                                                                                                  ; R
;  827     1121  2 IF .dif$gl_ignore [ign$v_comments]                              ! Ignoring comments?
;  828     1122  3    THEN BEGIN                                                   ! Then look for comments
;  829     1123  3         LOCAL index, bestptr, currptr;                          !                                                                  ; 1
;  830     1124  3         index = 0;                                              ! Init comment char count                                          ; 1
;  831     1125  3         bestptr = .charptr + .working_len;                      ! Point to end of string                                           ; 1
;  832     1126  4         WHILE (.index NEQ .dif$gl_commdesc [dsc$w_length])      ! Check for each comment character
;  833     1127  4         DO BEGIN                                                !
;  834     1128  5            IF (currptr = CH$FIND_SUB (.working_len, .charptr,   ! Yes, does comment char appear in record?
;  835     1129  4                 1, .dif$gl_commdesc [dsc$a_pointer] + .index)) NEQ 0
;  836     1130  4                THEN IF NOT .dif$gl_commflgs [.index]            ! Yes, must it appear in the first column?
;  837     1131  4                        THEN bestptr = MIN0 (.currptr, .bestptr) ! No, then we've definitely got a comment
;  838     1132  4                        ELSE IF .currptr EQL rdb [rdb$t_text]    ! Yes, then check that it does                                     ;
;  839     1133  4                                THEN EXITLOOP bestptr = .currptr; ! It does, so treat it as a comment                               ;
;  840     1134  4            index = .index + 1;                                  ! Increment the comment char count                                 ;
;  841     1135  3            END;                                                 !                                                                  ;
;  842     1136  3         working_len = .bestptr - .charptr;                      ! Reset record length                                              ;
;  843     1137  2         END;                                                    !                                                                  ;
;  844     1138  2
;  845     1139  2 !
;  846     1140  2 ! Only enter character loop if FORM_FEED or SPACING is specified.                                                                     ;
;  847     1141  2 ! Test each character in record to see if it should be ignored.  Edit ignored                                                         ;
;  848     1142  2 ! characters out of the record.                                                                                                       ;
;  849     1143  2 !                                                                                                                                    ;
;  850     1144  2 IF .dif$gl_ignore [ign$v_formfeed] OR .dif$gl_ignore [ign$v_spacing]                                                                 ;
;  851     1145  3 THEN BEGIN                                                                                                                           ;
;  852     1146  3                                                                                                                                      ;
;  853     1147  3      blankseen = false;                                         ! Init state                                                       ;
;  854     1148  3      ignore = false;                                            !                                                                  ;
;  855     1149  3      moveptr = .charptr;                                        !                                                                  ;
;  856     1150  3                                                                 !                                                                  ;
;  857     1151  4      WHILE (.charptr NEQ (rdb [rdb$t_text] + .working_len))     ! Check each character                                             ;
;  858     1152  4      DO BEGIN                                                   !                                                                  ;
;  859     1153  4         SELECTONE .charptr [0] OF SET                           !                                                                  ;
;  860     1154  4                                                                 !                                                                  ;
;  861     1155  4         [%X'0C']:                                               ! Form feed?                                                       ;
;  862     1156  4                 IF .dif$gl_ignore [ign$v_formfeed]              ! Are we ignoring them?                                            ;
;  863     1157  4                    THEN ignore = true                           ! Yes, then set ignore flag                                        ;
;  864     1158  4                    ELSE blankseen = false;                      ! Reset blank flag                                                 ; S
;  865     1159  4                                                                 !                                                                  ; R
;  866     1160  4         [%X'20',%X'09']:                                        ! Blank or tab?                                                    ; E
;  867     1161  4                 IF .dif$gl_ignore [ign$v_spacing]               ! Are we evening out spacing                                       ; L
;  868     1162  5                    THEN (IF .blankseen                          ! Yes                                                              ; L
;  869     1163  5                             THEN ignore = true                  ! Second blank or tab, so ignore it                                ; M
```

```
;  870          1164  6                          ELSE BEGIN                                      ! First blank or tab
;  871          1165  6                               blankseen = true;                          ! Set blankseen flag
;  872          1166  6                               IF .charptr [0] EQL %X'09'                 ! If char is a tab, we have
;  873          1167  7                               THEN BEGIN
;  874          1168  7                                   rdb [rdb$v_edited] = true;             ! editted the line
;  875          1169  7                                   charptr [0] = %X'20';                  ! Convert tabs to blanks
;  876          1170  6                                   END;
;  877          1171  4                               END);                                      !
;  878          1172  4
;  879          1173  4          [OTHERWISE]:                                                    ! All other characters
;  880          1174  4                  blankseen = false;                                      !
;  881          1175  4
;  882          1176  4          TES;
;  883          1177  4
;  884          1178  4 !
;  885          1179  4 ! If not ignoring last character tested, then copy the character.
;  886          1180  4 !
;  887          1181  4          IF NOT .ignore                                                  ! If last char was not ignored
;  888          1182  5             THEN BEGIN                                                   !
;  889          1183  5                  moveptr [0] = .charptr [0];                            ! Copy the character
;  890          1184  5                  moveptr = .moveptr + 1;                                ! Update result ptr
;  891          1185  4                  END;                                                   !
;  892          1186  4
;  893          1187  4          ignore = false;                                                 ! Reset flag
;  894          1188  4          charptr = .charptr + 1;                                         ! Incr the charptr
;  895          1189  3          END;                                                            ! Of while
;  896          1190  3
;  897          1191  3       working_len = .moveptr - rdb [rdb$t_text];                         ! Update working length
;  898          1192  2       END;
;  899          1193  2
;  900          1194  2 !
;  901          1195  2 ! If ignoring trailing blanks, then edit them out also.
;  902          1196  2 !
;  903          1197  2 moveptr = rdb [rdb$t_text] + .working_len - 1;                           ! Update move ptr
;  904          1198  2 IF .dif$gl_ignore [ign$v_traiblnk]                                       ! If ignoring trailing blanks
;  905          1199  3    THEN BEGIN                                                           !
;  906          1200  3         BIND start_of_record = rdb [rdb$t_text] - 1;
;  907          1201  4         WHILE ( (.moveptr [0] EQL %X'20') OR                           ! Then while last char is blank
;  908          1202  3                 (.moveptr [0] EQL %X'09') ) AND                        ! or tab
;  909          1203  4                 (.moveptr NEQ start_of_record)                         ! and not past beginning of string
;  910          1204  3         DO moveptr = .moveptr - 1;                                     ! Decrement size of record
;  911          1205  2         END;                                                           !
;  912          1206  2
;  913          1207  2 !
;  914          1208  2 ! Calculate length of edited record.  Set edited flag if different from original length
;  915          1209  2 !
;  916          1210  2 working_len = .rdb [rdb$w_length];
;  917          1211  2 rdb [rdb$w_length] = .moveptr - rdb [rdb$t_text] + 1;
;  918          1212  2 IF .working_len NEQ .rdb [rdb$w_length]
;  919          1213  2    THEN rdb [rdb$v_edited] = true;
;  920          1214  2
;  921          1215  2 !
;  922          1216  2 ! If length is now zero, and we are ignoring blank records, then ignore
;  923          1217  2 ! this record.
;  924          1218  2 !
;  925          1219  2 IF .dif$gl_ignore [ign$v_blnklin] AND (.rdb [rdb$w_length] EQL 0)
;  926          1220  2    THEN rdb [rdb$v_ignored] = rdb [rdb$v_edited] = true;
```

```
;  927          1221  2
;  928          1222  2 RETURN true;
;  929          1223  1 END;                                                    ! Of process_record


                                    0FFC 00000 PROCESS_RECORD:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                    ; 1050
                      5B 00000000G   00   9E 00002          MOVAB   DIF$GL_IGNORE, R11                          ;
                      50       04    AC   D0 00009          MOVL    FDB, R0                                     ; 1088
                      54       08    A0   D0 0000D          MOVL    8(R0), RDB                                  ; 1089
                      59       14    A4   9E 00011          MOVAB   20(R4), R9                                  ; 1090
                      58             59   D0 00015          MOVL    R9, CHARPTR                                 ; 1095
                      5A       12    A4   9E 00018          MOVAB   18(RDB), R10                                ; 1096
                      57             6A   3C 0001C          MOVZWL  (R10), WORKING_LEN                          ;
                  2A  6B             05   E1 0001F          BBC     #5, DIF$GL_IGNORE, 4$                       ; 1099
           00000000G  00       28    A0   D1 00023          CMPL    40(R0), DIF$GL_HEADER                       ; 1100
                                   05   1E 0002B          BGEQU   1$                                          ; 1104
                              28    A0   D6 0002D          INCL    40(R0)                                      ; 1105
                                   18   11 00030          BRB     3$                                          ; 1107
                                   57   D5 00032 1$:      TSTL    WORKING_LEN                                 ; 1108
                                   17   13 00034          BEQL    4$                                          ; 1110
                      0C                68   91 00036          CMPB    (CHARPTR), #12                          ; 1111
                                   12   12 00039          BNEQ    4$                                          ; 1121
                      01                57   D1 0003B          CMPL    WORKING_LEN, #1                         ; 1124
                                   06   12 0003E          BNEQ    2$                                          ; 1125
                                  28    A0   D4 00040          CLRL    40(R0)                                  ; 1126
                                 00FF   31   00043          BRW     28$                                         ; 1129
                             28    A0         01   D0 00046 2$:      MOVL    #1, 40(R0)                      ;
                                 00F4   31   0004A 3$:      BRW     27$                                         ;
                  52                68         01   E1 0004D 4$:      BBC     #1, DIF$GL_IGNORE, 11$          ; 1130
                                   56   D4 00051          CLRL    INDEX                                       ; 1131
                  55                58         57   C1 00053          ADDL3   WORKING_LEN, CHARPTR, BESTPTR   ;
 56 00000000G     00                10         00   ED 00057 5$:      CMPZV   #0, #16, DIF$GL_COMMDESC, INDEX ; 1132
                                   3D   13 00060          BEQL    10$                                         ;
                      50 00000000G   00   D0 00062          MOVL    DIF$GL_COMMDESC+4, R0                   ; 1133
 68               57           6640         01   39 00069          MATCHC  #1, (INDEX)[R0], WORKING_LEN, (CHARPTR)
                                   03   13 0006F          BEQL    6$                                          ; 1134
                              53         01   D0 00071          MOVL    #1, R3                                  ; 1136
                              51         73   9E 00074 6$:      MOVAB   -(R3), CURRPTR                          ;
                                   22   13 00077          BEQL    9$                                          ;
                  10 00000000G   00         56   E0 00079          BBS     INDEX, DIF$GL_COMMFLGS, 8$      ; 1137
                              53         51   D0 00081          MOVL    CURRPTR, R3                             ; 1138
                              55         53   D1 00084          CMPL    R3, BESTPTR                             ;
                                   03   1B 00087          BLEQU   7$                                          ;
                              53         55   D0 00089          MOVL    BESTPTR, R3                             ;
                              55         53   D0 0008C 7$:      MOVL    R3, BESTPTR                             ;
                                   0A   11 0008F          BRB     9$                                          ;
                              59         51   D1 00091 8$:      CMPL    CURRPTR, R9                             ; 1139
                                   05   12 00094          BNEQ    9$                                          ;
                              55         51   D0 00096          MOVL    CURRPTR, BESTPTR                        ; 1140
                                   04   11 00099          BRB     10$                                         ;
                                   56   D6 0009B 9$:      INCL    INDEX                                       ; 1141
                                   B8   11 0009D          BRB     5$                                          ; 1142
                  57               55         58   C3 0009F 10$:     SUBL3   CHARPTR, BESTPTR, WORKING_LEN   ; 1143
```

```
55      6B          01     02  EF 000A3 11$:  EXTZV   #2, #1, DIF$GL_IGNORE, R5              ; 1144
                    04     55  E8 000A8       BLBS    R5, 12$                                ;
        52          6B     04  E1 000AB       BBC     #4, DIF$GL_IGNORE, 22$                 ;
                           51  7C 000AF 12$:  CLRQ    IGNORE                                 ; 1148
                    50     58  D0 000B1       MOVL    CHARPTR, MOVEPTR                       ; 1149
                    53  14 A744  9E 000B4     MOVAB   20(WORKING_LEN)[RDB], R3               ; 1151
                    53     58  D1 000B9 13$:  CMPL    CHARPTR, R3                            ;
                           3F  13 000BC       BEQL    21$                                    ;
                    0C     68  91 000BE       CMPB    (CHARPTR), #12                         ; 1155
                           05  12 000C1       BNEQ    14$                                    ;
                    13     55  E8 000C3       BLBS    R5, 16$                                ; 1156
                           27  11 000C6       BRB     18$                                    ; 1158
                    09     68  91 000C8 14$:  CMPB    (CHARPTR), #9                          ; 1160
                           05  13 000CB       BEQL    15$                                    ;
                    20     68  91 000CD       CMPB    (CHARPTR), #32                         ;
                           1D  12 000D0       BNEQ    18$                                    ;
        1B          6B     04  E1 000D2 15$:  BBC     #4, DIF$GL_IGNORE, 19$                 ; 1161
                    05     52  E9 000D6       BLBC    BLANKSEEN, 17$                         ; 1162
                    51     01  D0 000D9 16$:  MOVL    #1, IGNORE                             ; 1163
                           13  11 000DC       BRB     19$                                    ;
                    52     01  D0 000DE 17$:  MOVL    #1, BLANKSEEN                          ; 1165
                    09     68  91 000E1       CMPB    (CHARPTR), #9                          ; 1166
                           0B  12 000E4       BNEQ    19$                                    ;
              08    A4     08  88 000E6       BISB2   #8, 8(RDB)                             ; 1168
                    68     20  90 000EA       MOVB    #32, (CHARPTR)                         ; 1169
                           02  11 000ED       BRB     19$                                    ; 1162
                           52  D4 000EF 18$:  CLRL    BLANKSEEN                              ; 1174
                    03     51  E8 000F1 19$:  BLBS    IGNORE, 20$                            ; 1181
                    80     68  90 000F4       MOVB    (CHARPTR), (MOVEPTR)+                  ; 1183
                           51  D4 000F7 20$:  CLRL    IGNORE                                 ; 1187
                           58  D6 000F9       INCL    CHARPTR                                ; 1188
                           BC  11 000FB       BRB     13$                                    ; 1151
        57          50     59  C3 000FD 21$:  SUBL3   R9, MOVEPTR, WORKING_LEN               ; 1191
                    50  13 A744  9E 00101 22$: MOVAB  19(WORKING_LEN)[RDB], MOVEPTR          ; 1197
        17          6B     03  E1 00106       BBC     #3, DIF$GL_IGNORE, 25$                 ; 1198
                    20     60  91 0010A 23$:  CMPB    (MOVEPTR), #32                         ; 1201
                           05  13 0010D       BEQL    24$                                    ;
                    09     60  91 0010F       CMPB    (MOVEPTR), #9                          ; 1202
                           0D  12 00112       BNEQ    25$                                    ;
                    51  FF A9  9E 00114 24$:  MOVAB   -1(R9), R1                             ; 1203
                    51     50  D1 00118       CMPL    MOVEPTR, R1                            ;
                           04  13 0011B       BEQL    25$                                    ;
                           50  D7 0011D       DECL    MOVEPTR                                ; 1204
                           E9  11 0011F       BRB     23$                                    ;
                    57     6A  3C 00121 25$:  MOVZWL  (R10), WORKING_LEN                     ; 1210
                    50     59  C2 00124       SUBL2   R9, R0                                 ; 1211
        6A          50     01  A1 00127       ADDW3   #1, R0, (R10)                          ;
57      6A          10     00  ED 0012B       CMPZV   #0, #16, (R10), WORKING_LEN            ; 1212
                           04  13 00130       BEQL    26$                                    ;
              08    A4     08  88 00132       BISB2   #8, 8(RDB)                             ; 1213
                    0C     6B  E9 00136 26$:  BLBC    DIF$GL_IGNORE, 28$                     ; 1219
                           6A  B5 00139       TSTW    (R10)                                  ;
                           08  12 0013B       BNEQ    28$                                    ;
              08    A4     08  88 0013D       BISB2   #8, 8(RDB)                             ; 1220
              08    A4     01  88 00141 27$:  BISB2   #1, 8(RDB)                             ;
                    50     01  D0 00145 28$:  MOVL    #1, R0                                 ; 1222
                               04 00148       RET                                            ; 1223
```

; Routine Size: 329 bytes,   Routine Base: $CODE$ + 0533

```
;  931          1224  1 ROUTINE allocate_rdb (rdbaddr, textlen) =
;  932          1225  2 BEGIN
;  933          1226  2
;  934          1227  2 !++
;  935          1228  2 !
;  936          1229  2 ! FUNCTIONAL DESCRIPTION:
;  937          1230  2 !
;  938          1231  2 !       This routine is called to allocate a new RDB.
;  939          1232  2 !
;  940          1233  2 ! INPUTS:
;  941          1234  2 !
;  942          1235  2 !       rdbaddr =  The address of a longword to receive the address of the
;  943          1236  2 !                  newly allocated RDB.
;  944          1237  2 !
;  945          1238  2 !       textlen =  The length of the text that will make up the variably
;  946          1239  2 !                  sized portion of the RDB.
;  947          1240  2 !
;  948          1241  2 ! OUTPUTS:
;  949          1242  2 !
;  950          1243  2 !       The new RDB is allocated and its fields are all zeroed.
;  951          1244  2 !
;  952          1245  2 ! ROUTINE VALUES:
;  953          1246  2 !
;  954          1247  2 !       Always true.
;  955          1248  2 !
;  956          1249  2 !--
;  957          1250  2
;  958          1251  2 LOCAL
;  959          1252  2     rdb : REF BBLOCK;
;  960          1253  2
;  961          1254  2 IF NOT LIB$GET_VM (%REF (.textlen + rdb$c_size), rdb)          ! Allocate RDB
;  962          1255  2    THEN RETURN dif$_insvirmem;
;  963          1256  2
;  964          1257  2 CH$FILL (%X'00', rdb$c_size, .rdb);                          ! Init it
;  965          1258  2 .rdbaddr = .rdb;                                             ! Return its address
;  966          1259  2
;  967          1260  2 RETURN true;
;  968          1261  1 END;


                                        003C 00000 ALLOCATE_RDB:
                                                   .WORD   Save R2,R3,R4,R5                    ; 1224
                                5E              08 C2 00002         SUBL2   #8, SP
                                            04  AE 9F 00005         PUSHAB  RDB                 ; 1254
                04  AE      08 AC              14 C1 00008         ADDL3   #20, TEXTLEN, 4(SP)
                                            04  AE 9F 0000E         PUSHAB  4(SP)
                    00000000G   00              02 FB 00011         CALLS   #2, LIB$GET_VM
                                08              50 E8 00018         BLBS    R0, 1$
                                50 00000000G    8F D0 0001B         MOVL    #DIF$_INSVIRMEM, R0 ; 1255
                                                04 00022         RET
14                  00          6E              00 2C 00023 1$:     MOVC5   #0, (SP), #0, #20, @RDB ; 1257
                                            04  BE    00028
                        04  BC          04  AE D0 0002A         MOVL    RDB, @RDBADDR,          ; 1258
                                50              01 D0 0002F         MOVL    #1, R0              ; 1260
```

```
                04 00032        RET                                             ; 1261

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 067C
```

```
;  970          1262  1 ROUTINE purge_rdb (fdb) =
;  971          1263  2 BEGIN
;  972          1264  2
;  973          1265  2 !++
;  974          1266  2 !
;  975          1267  2 ! FUNCTIONAL DESCRIPTION:
;  976          1268  2 !
;  977          1269  2 !       This routine is called to purge the RDB's associated
;  978          1270  2 !       with a particular file.
;  979          1271  2 !
;  980          1272  2 ! INPUTS:
;  981          1273  2 !
;  982          1274  2 !       fdb =   The address of the FDB for the file whose RDB's are to be
;  983          1275  2 !               purged.  CURREC specifies the first RDB not to be purged.
;  984          1276  2 !
;  985          1277  2 ! OUTPUTS:
;  986          1278  2 !
;  987          1279  2 !       The purged RDB's are deallocated, and the FIRSTREC field of the
;  988          1280  2 !       FDB is updated.
;  989          1281  2 !
;  990          1282  2 ! ROUTINE VALUES:
;  991          1283  2 !
;  992          1284  2 !       Always true
;  993          1285  2 !
;  994          1286  2 !--
;  995          1287  2
;  996          1288  2 MAP
;  997          1289  2     fdb : REF BBLOCK;
;  998          1290  2
;  999          1291  2 LOCAL
; 1000          1292  2     rdb : REF BBLOCK;
; 1001          1293  2
; 1002          1294  2 IF (rdb = .fdb [fdb$l_firstrec]) EQL .fdb [fdb$l_currec]
; 1003          1295  2    THEN RETURN true
; 1004          1296  3    ELSE BEGIN
; 1005          1297  3         fdb [fdb$l_firstrec] = .rdb [rdb$l_flink];
; 1006          1298  3         IF NOT .rdb [rdb$v_permanent]
; 1007          1299  3            THEN LIB$FREE_VM ( %REF(.rdb [rdb$w_length] + rdb$c_size), rdb);
; 1008          1300  3         RETURN purge_rdb (.fdb);
; 1009          1301  2         END;
; 1010          1302  2
; 1011          1303  1 END;


                                0004 00000 PURGE_RDB:
                                           .WORD   Save R2                         ;  1262
                    5E          08   C2 00002          SUBL2   #8, SP              ;
                    52    04    AC   D0 00005          MOVL    FDB, R2             ;  1294
                    50    04    A2   D0 00009          MOVL    4(R2), R0           ;
                 04 AE          50   D0 0000D          MOVL    R0, RDB             ;
                    62          50   D1 00011          CMPL    R0, (R2)            ;
                                04   12 00014          BNEQ    1$                  ;
                    50          01   D0 00016          MOVL    #1, R0              ;  1296
                                     04 00019          RET                         ;
```

```
                          50      04   AE  D0 0001A 1$:    MOVL    RDB, R0                                 ;  1297
                       04 A2          60  D0 0001E         MOVL    (R0), 4(R2)                             ;
                    16 08 A0          01  E0 00022         BBS     #1, 8(R0), 2$                           ;  1298
                                   04 AE  9F 00027         PUSHAB  RDB                                     ;  1299
                       04 AE       12 A0  3C 0002A         MOVZWL  18(R0), 4(SP)                           ;
                       04 AE          14  C0 0002F         ADDL2   #20, 4(SP)                              ;
                                   04 AE  9F 00033         PUSHAB  4(SP)                                   ;
                00000000G  00          02  FB 00036         CALLS   #2, LIB$FREE_VM                         ;
                                       52  DD 0003D 2$:     PUSHL   R2                                      ;  1300
                       BD AF           01  FB 0003F         CALLS   #1, PURGE_RDB                           ;
                                          04 00043         RET                                             ;  1303

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 06AF
```

```
;  1013          1304  1 ROUTINE set_move_flags =
;  1014          1305  2 BEGIN
;  1015          1306  2
;  1016          1307  2 !++
;  1017          1308  2 !
;  1018          1309  2 ! FUNCTIONAL DESCRIPTION:
;  1019          1310  2 !
;  1020          1311  2 !       This routine is called to initialize the FDB move flags.
;  1021          1312  2 !       An FDB move flag is set to true if that file will be output
;  1022          1313  2 !       in more than one format or radix.  Both FDB move flags are
;  1023          1314  2 !       false if SLP output has been specified.
;  1024          1315  2 !
;  1025          1316  2 ! INPUTS:
;  1026          1317  2 !
;  1027          1318  2 !       None.                                                                   ; R
;  1028          1319  2 !
;  1029          1320  2 ! OUTPUTS:
;  1030          1321  2 !
;  1031          1322  2 !       None.
;  1032          1323  2 !
;  1033          1324  2 ! ROUTINE VALUES:
;  1034          1325  2 !
;  1035          1326  2 !       Always true.
;  1036          1327  2 !
;  1037          1328  2 !--
;  1038          1329  2
;  1039          1330  2 LOCAL
;  1040          1331  2     multiradix;
;  1041          1332  2
;  1042          1333  2 IF .dif$gl_flags [dif$v_slp]
;  1043          1334  2     THEN RETURN true;
;  1044          1335  2
;  1045          1336  3 IF (.dif$gl_flags [dif$v_ascii] + .dif$gl_flags [dif$v_hex] +
;  1046          1337  2      .dif$gl_flags [dif$v_octal]) GTR 1
;  1047          1338  3     THEN BEGIN
;  1048          1339  3          multiradix = true;
;  1049          1340  3          IF .dif$gl_flags [dif$v_merged]
;  1050          1341  4             THEN BEGIN
;  1051          1342  4                  dif$gl_masfdb [fdb$v_move] = true;
;  1052          1343  4                  dif$gl_revfdb [fdb$v_move] = true;
;  1053          1344  4                  RETURN true;
;  1054          1345  3                  END;
;  1055          1346  3          END
;  1056          1347  2     ELSE multiradix = false;
;  1057          1348  2
;  1058          1349  4 IF (((.multiradix OR .dif$gl_flags [dif$v_parallel]) +
;  1059          1350  4          .dif$gl_flags [dif$v_merged] +
;  1060          1351  4          .dif$gl_masfdb [fdb$v_separated] +
;  1061          1352  2          .dif$gl_masfdb [fdb$v_changebar]) GTR 1) OR
;  1062          1353  3     (.dif$gl_masfdb [fdb$v_changebar] AND .dif$gl_revfdb [fdb$v_separated])
;  1063          1354  2     THEN dif$gl_masfdb [fdb$v_move] = true;
;  1064          1355  2
;  1065          1356  4 IF (((.multiradix OR .dif$gl_flags [dif$v_parallel]) +
;  1066          1357  4          .dif$gl_flags [dif$v_merged] +
;  1067          1358  4          .dif$gl_revfdb [fdb$v_separated] +
;  1068          1359  2          .dif$gl_revfdb [fdb$v_changebar]) GTR 1) OR
;  1069          1360  4     ((.dif$gl_masfdb [fdb$v_separated] +
```

```
;  1070          1361  4          .dif$gl_revfdb [fdb$v_separated] +
;  1071          1362  2          .dif$gl_revfdb [fdb$v_changebar]) GTR 1) OR
;  1072          1363  3     (.dif$gl_masfdb [fdb$v_changebar] AND .dif$gl_revfdb [fdb$v_changebar])
;  1073          1364  2     THEN dif$gl_revfdb [fdb$v_move] = true;
;  1074          1365  2
;  1075          1366  2 RETURN true;
;  1076          1367  1 END;


                                                003C 00000 SET_MOVE_FLAGS:
                                                           .WORD   Save R2,R3,R4,R5                        ; 1304
                                     55 00000000G   00  9E 00002           MOVAB   DIF$GL_REVFDB+36, R5
                                     54 00000000G   00  9E 00009           MOVAB   DIF$GL_MASFDB+36, R4
                                     53 00000000G   00  9E 00010           MOVAB   DIF$GL_FLAGS, R3
                                     03        01   A3  E9 00017           BLBC    DIF$GL_FLAGS+1, 1$      ; 1333
                                               0085  31 0001B             BRW     7$
                50              63        01        00  EF 0001E 1$:      EXTZV   #0, #1, DIF$GL_FLAGS, R0 ; 1336
                51              63        01        01  EF 00023          EXTZV   #1, #1, DIF$GL_FLAGS, R1
                                          50        51  C0 00028          ADDL2   R1, R0
                52              63        01        02  EF 0002B          EXTZV   #2, #1, DIF$GL_FLAGS, R2 ; 1337
                                          50        52  C0 00030          ADDL2   R2, R0
                                          01        50  D1 00033          CMPL    R0, #1
                                                    0C  15 00036          BLEQ    2$
                                          50        01  D0 00038          MOVL    #1, MULTIRADIX          ; 1339
                                07        63        05  E1 0003B          BBC     #5, DIF$GL_FLAGS, 3$    ; 1340
                                          64        08  88 0003F          BISB2   #8, DIF$GL_MASFDB+36    ; 1342
                                                    5C  11 00042          BRB     6$                      ; 1343
                                                    50  D4 00044 2$:       CLRL    MULTIRADIX              ; 1347
                51              63        01        06  EF 00046 3$:      EXTZV   #6, #1, DIF$GL_FLAGS, R1 ; 1349
                                          50        51  C8 0004B          BISL2   R1, R0
                51              63        01        05  EF 0004E          EXTZV   #5, #1, DIF$GL_FLAGS, R1 ; 1350
                                          50        51  C0 00053          ADDL2   R1, R0                   ; 1349
                51              64        01        02  EF 00056          EXTZV   #2, #1, DIF$GL_MASFDB+36, R1 ; 1351
                                          51        50  C0 0005B          ADDL2   R0, R1                   ; 1350
                52              64        01        00  EF 0005E          EXTZV   #0, #1, DIF$GL_MASFDB+36, R2 ; 1352
                                          51        52  C0 00063          ADDL2   R2, R1
                                          01        51  D1 00066          CMPL    R1, #1
                                                    07  14 00069          BGTR    4$
                                          07        52  E9 0006B          BLBC    R2, 5$                   ; 1353
                                03        65        02  E1 0006E          BBC     #2, DIF$GL_REVFDB+36, 5$
                                          64        08  88 00072 4$:       BISB2   #8, DIF$GL_MASFDB+36     ; 1354
                52              65        01        02  EF 00075 5$:      EXTZV   #2, #1, DIF$GL_REVFDB+36, R2 ; 1356
                                          50        52  C0 0007A          ADDL2   R2, R0                   ; 1355
                51              65        01        00  EF 0007D          EXTZV   #0, #1, DIF$GL_REVFDB+36, R1 ; 1357
                                          50        51  C0 00082          ADDL2   R1, R0                   ; 1356
                                          01        50  D1 00085          CMPL    R0, #1
                                                    16  14 00088          BGTR    6$
                50              64        01        02  EF 0008A          EXTZV   #2, #1, DIF$GL_MASFDB+36, R0 ; 1360
                                          50        52  C0 0008F          ADDL2   R2, R0
                                          50        51  C0 00092          ADDL2   R1, R0                   ; 1362
                                          01        50  D1 00095          CMPL    R0, #1
                                                    06  14 00098          BGTR    6$
                                          06        64  E9 0009A          BLBC    DIF$GL_MASFDB+36, 7$     ; 1363
                                          03        51  E9 0009D          BLBC    R1, 7$
```

```
                65              08  88 000A0 6$:    BISB2   #8, DIF$GL_REVFDB+36                    ; 1364
                50              01  D0 000A3 7$:    MOVL    #1, R0                                  ; 1366
                                    04 000A6        RET                                             ; 1367

; Routine Size:  167 bytes,    Routine Base:  $CODE$ + 06F3
```

```
;  1078          1368  1 ROUTINE print_and_quit (difreccnt, status) =
;  1079          1369  2 BEGIN
;  1080          1370  2
;  1081          1371  2 !++
;  1082          1372  2 !
;  1083          1373  2 ! FUNCTIONAL DESCRIPTION:
;  1084          1374  2 !
;  1085          1375  2 !       This routine is called to print whatever records have been
;  1086          1376  2 !       processed when a fatal error occurs.
;  1087          1377  2 !
;  1088          1378  2 ! INPUTS:
;  1089          1379  2 !
;  1090          1380  2 !       difreccnt =     Count of last set of difference records.
;  1091          1381  2 !
;  1092          1382  2 !       status =        Status of error causing diff to quit.
;  1093          1383  2 !
;  1094          1384  2 ! OUTPUTS:
;  1095          1385  2 !
;  1096          1386  2 !       None.
;  1097          1387  2 !
;  1098          1388  2 ! ROUTINE VALUES:
;  1099          1389  2 !
;  1100          1390  2 !       The input status is returned.
;  1101          1391  2 !
;  1102          1392  2 !--
;  1103          1393  2 LOCAL
;  1104          1394  2     rdb : REF BBLOCK;
;  1105          1395  2
;  1106          1396  2 dif$gl_difrec = .dif$gl_difrec + (.difreccnt-1)/2 + 1;  ! Update difference count
;  1107          1397  2
;  1108          1398  2 dif$gl_merged = dif$gl_parallel = 0;                    ! Minimize no. of matched records to output
;  1109          1399  2
;  1110          1400  2 rdb = .dif$gl_masfdb [fdb$l_currec];                   ! Set up master FDB for output
;  1111          1401  2 dif$gl_masfdb [fdb$l_compnrec] = .rdb;
;  1112          1402  2 rdb [rdb$l_flink] = .dif$gl_masfdb [fdb$l_eofrec];
;  1113          1403  2
;  1114          1404  2 rdb = .dif$gl_revfdb [fdb$l_currec];                   ! Set up revision FDB for output
;  1115          1405  2 dif$gl_revfdb [fdb$l_compnrec] = .rdb;
;  1116          1406  2 rdb [rdb$l_flink] = .dif$gl_revfdb [fdb$l_eofrec];
;  1117          1407  2
;  1118          1408  2 write_mismatch ();                                     ! Output differences
;  1119          1409  2
;  1120          1410  2 RETURN .status;
;  1121          1411  1 END;


                                001C 00000 PRINT_AND_QUIT:
                                           .WORD   Save R2,R3,R4                                   ; 1368
                    54 00000000G   00  9E 00002    MOVAB   DIF$GL_DIFREC, R4
                    53 00000000G   00  9E 00009    MOVAB   DIF$GL_REVFDB, R3
                    52 00000000G   00  9E 00010    MOVAB   DIF$GL_MASFDB, R2
                    50             64  D0 00017    MOVL    DIF$GL_DIFREC, R0
                                                                                                   ; 1396
          51        04   AC        01  C3 0001A    SUBL3   #1, DIFRECCNT, R1
                         51        02  C6 0001F    DIVL2   #2, R1
```

```
                        64       01 A140  9E 00022         MOVAB   1(R1)[R0], DIF$GL_DIFREC                  ;
                    00000000G    00       D4 00027         CLRL    DIF$GL_PARALLEL                          ; 1398
                    00000000G    00       D4 0002D         CLRL    DIF$GL_MERGED                            ;
                        50               62 D0 00033      MOVL    DIF$GL_MASFDB, RDB                       ; 1400
                    14  A2               50 D0 00036      MOVL    RDB, DIF$GL_MASFDB+20                    ; 1401
                        60       18      A2 D0 0003A      MOVL    DIF$GL_MASFDB+24, (RDB)                  ; 1402
                        50               63 D0 0003E      MOVL    DIF$GL_REVFDB, RDB                       ; 1404
                    14  A3               50 D0 00041      MOVL    RDB, DIF$GL_REVFDB+20                    ; 1405
                        60       18      A3 D0 00045      MOVL    DIF$GL_REVFDB+24, (RDB)                  ; 1406
                00000000G    00           00 FB 00049      CALLS   #0, WRITE_MISMATCH                       ; 1408
                        50       08      AC D0 00050      MOVL    STATUS, R0                               ; 1410
                                             04 00054      RET                                              ; 1411

; Routine Size:  85 bytes,    Routine Base:  $CODE$ + 079A

; 1122          1412  1
; 1123          1413  1 END                                                   ! Of module
; 1124          1414  0 ELUDOM



                                                              .EXTRN  LIB$STOP

;                               PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
;  $CODE$                                2031  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]STARLET.L32;1              9776        17         0       581          00:01.0



;                                       COMMAND QUALIFIERS

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

; Size:         2031 code + 0 data bytes
; Run Time:       00:36.9
; Elapsed Time:   01:19.5
; Lines/CPU Min:   2302
; Lexemes/CPU-Min: 19826
; Memory Used:  187 pages
```

```
; Compilation Complete
```

DIFMSG
LIS

DIRECTORY
LIS

MAIN
LIS

DIR

DIRECTORY
MAP

DIRECTDEF
REQ

OUTPUT
LIS

DISPLYDEF
SDL

DIRECTMSG
LIS