J 4

```
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
```

```
RRRRRRRR    MM      MM    SSSSSSSS    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  BBBBBBBB
RRRRRRRR    MM      MM    SSSSSSSS    CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  BBBBBBBB
RR      RR  MMMM  MMMM  SS          CC          HH      HH  EE          CC          KK      KK  BB      BB
RR      RR  MMMM  MMMM  SS          CC          HH      HH  EE          CC          KK      KK  BB      BB
RR      RR  MM  MM  MM  SS          CC          HH      HH  EE          CC          KK    KK    BB      BB
RR      RR  MM  MM  MM  SS          CC          HH      HH  EE          CC          KK    KK    BB      BB
RRRRRRRR    MM      MM    SSSSSS    CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKKK      BBBBBBBB
RRRRRRRR    MM      MM    SSSSSS    CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKKK      BBBBBBBB
RR  RR      MM      MM          SS  CC          HH      HH  EE          CC          KK    KK    BB      BB
RR  RR      MM      MM          SS  CC          HH      HH  EE          CC          KK    KK    BB      BB
RR    RR    MM      MM          SS  CC          HH      HH  EE          CC          KK      KK  BB      BB      ....
RR    RR    MM      MM          SS  CC          HH      HH  EE          CC          KK      KK  BB      BB      ....
RR      RR  MM      MM  SSSSSSSS      CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  BBBBBBBB        ....
RR      RR  MM      MM  SSSSSSSS      CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  BBBBBBBB        ....

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 %title 'RMSCHECKB - Check a File Structure'
;  2     0002  0         module rmscheckb (
;  3     0003  1                           ident='V04-000') = begin
;  4     0004  1
;  5     0005  1 !
;  6     0006  1 !*****************************************************************************
;  7     0007  1 !*                                                                           *
;  8     0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;  9     0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
; 10     0010  1 !*  ALL RIGHTS RESERVED.                                                     *
; 11     0011  1 !*                                                                           *
; 12     0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 13     0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 14     0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 15     0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
; 16     0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
; 17     0017  1 !*  TRANSFERRED.                                                             *
; 18     0018  1 !*                                                                           *
; 19     0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 20     0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 21     0021  1 !*  CORPORATION.                                                             *
; 22     0022  1 !*                                                                           *
; 23     0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 24     0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 25     0025  1 !*                                                                           *
; 26     0026  1 !*                                                                           *
; 27     0027  1 !*****************************************************************************
; 28     0028  1 !
; 29     0029  1
; 30     0030  1 !++
; 31     0031  1 ! Facility:     VAX/VMS Analyze Facility, Check a File Structure
; 32     0032  1 !
; 33     0033  1 ! Abstract:     This module is responsible for checking the structure of
; 34     0034  1 !               an RMS file as requested via /CHECK.  It also prepares a
; 35     0035  1 !               report of the results.
; 36     0036  1 !
; 37     0037  1 !
; 38     0038  1 ! Environment:
; 39     0039  1 !
; 40     0040  1 ! Author: Paul C. Anagnostopoulos, Creation Date: 5 August 1981
; 41     0041  1 !
; 42     0042  1 ! Modified By:
; 43     0043  1 !
; 44     0044  1 !       V03-004 DGB0048         Donald G. Blair         08-May-1984
; 45     0045  1 !               Fix condition handling so ANALYZRMS returns the correct
; 46     0046  1 !               error status at image exit.  Change condition handler
; 47     0047  1 !               from ANL$CONDITION_HANDLER to ANL$UNWIND_HANDLER.
; 48     0048  1 !
; 49     0049  1 !       V03-003 PCA1011         Paul C. Anagnostopoulos  1-Apr-1983
; 50     0050  1 !               Change the message prefix to ANLRMS$_ to ensure that
; 51     0051  1 !               message symbols are unique across all ANALYZEs.  This
; 52     0052  1 !               is necessitated by the new merged message files.
; 53     0053  1 !
; 54     0054  1 !       V03-002 PCA1001         Paul C. Anagnostopoulos  5-Nov-1982
; 55     0055  1 !               Add code to support the new /SUMMARY mode.
; 56     0056  1 !
; 57     0057  1 !       V03-001 PCA0062         Paul Anagnostopoulos    29-Mar-1982
```

```
;   58          0058  1 !
;   59          0059  1 !                   Fix bug in code that determines when the analysis of
;   60          0060  1 !                   indexed file data blocks is complete.  It was skipping
;   61          0061  1 !--                 some blocks at times.
```

```
;    63          0062  1 %sbttl 'Module Declarations'
;    64          0063  1 !
;    65          0064  1 ! Libraries and Requires:
;    66          0065  1 !
;    67          0066  1
;    68          0067  1 library 'lib';
;    69          0068  1 require 'rmsreq';
;    70          0577  1
;    71          0578  1 !
;    72          0579  1 ! Table of Contents:
;    73          0580  1 !
;    74          0581  1
;    75          0582  1 forward routine
;    76          0583  1         anl$idx_check: novalue,
;    77          0584  1         anl$idx_check_key_stuff: novalue;
;    78          0585  1
;    79          0586  1 !
;    80          0587  1 ! External References:
;    81          0588  1 !
;    82          0589  1
;    83          0590  1 external routine
;    84          0591  1         anl$area_descriptor,
;    85          0592  1         anl$area_statistics,
;    86          0593  1         anl$bucket,
;    87          0594  1         anl$2bucket_header,
;    88          0595  1         anl$3bucket_header,
;    89          0596  1         anl$format_error,
;    90          0597  1         anl$format_line,
;    91          0598  1         anl$format_skip,
;    92          0599  1         anl$idx_prolog,
;    93          0600  1         anl$2index_record,
;    94          0601  1         anl$3index_record,
;    95          0602  1         anl$key_descriptor,
;    96          0603  1         anl$key_statistics,
;    97          0604  1         anl$unwind_handler,
;    98          0605  1         anl$2primary_data_record,
;    99          0606  1         anl$3primary_data_record,
;   100          0607  1         anl$prolog_checksums,
;   101          0608  1         anl$3reclaimed_bucket_header,
;   102          0609  1         anl$2sidr_record,
;   103          0610  1         anl$3sidr_record,
;   104          0611  1         lib$establish: addressing_mode(general);
;   105          0612  1
;   106          0613  1 external
;   107          0614  1         anl$gb_mode: byte,
;   108          0615  1         anl$gl_fat: ref block[,byte],
;   109          0616  1         anl$gw_prolog: word;
;   110          0617  1
;   111          0618  1 !
;   112          0619  1 ! Own Variables:
;   113          0620  1 !
```

```
;  115          0621  1 %sbttl 'ANL$IDX_CHECK - Check Structure of Indexed File'
;  116          0622  1 !++
;  117          0623  1 ! Functional Description:
;  118          0624  1 !       This routine is responsible for checking the structure of an
;  119          0625  1 !       indexed file, as requested by /CHECK mode.
;  120          0626  1 !
;  121          0627  1 !       It is also responsible for producing the statistics requested
;  122          0628  1 !       by /STATISTICS mode.  This is done as a superset of /CHECK mode,
;  123          0629  1 !       so the structure gets checked while the statistics are done.
;  124          0630  1 !
;  125          0631  1 ! Formal Parameters:
;  126          0632  1 !       none
;  127          0633  1 !
;  128          0634  1 ! Implicit Inputs:
;  129          0635  1 !       global data
;  130          0636  1 !
;  131          0637  1 ! Implicit Outputs:
;  132          0638  1 !       global data
;  133          0639  1 !
;  134          0640  1 ! Returned Value:
;  135          0641  1 !       none
;  136          0642  1 !
;  137          0643  1 ! Side Effects:
;  138          0644  1 !
;  139          0645  1 !--
;  140          0646  1
;  141          0647  1
;  142          0648  2 global routine anl$idx_check: novalue = begin
;  143          0649  2
;  144          0650  2 local
;  145          0651  2         p: bsd, c: bsd,
;  146          0652  2         sp: ref block[,byte],
;  147          0653  2         area_count: long,
;  148          0654  2         id: long,
;  149          0655  2         areas_vector: vector[256,byte],
;  150          0656  2         another: byte;
;  151          0657  2
;  152          0658  2
;  153          0659  2 ! Establish the condition handler for drastic structure errors.
;  154          0660  2
;  155          0661  2 lib$establish(anl$unwind_handler);
;  156          0662  2
;  157          0663  2 ! First we want to check the checksums in the prolog blocks.
;  158          0664  2
;  159          0665  2 anl$prolog_checksums();
;  160          0666  2
;  161          0667  2 ! Now we can read in the first prolog block and check the fixed portion
;  162          0668  2 ! of the prolog.
;  163          0669  2
;  164          0670  2 init_bsd(p);
;  165          0671  2 p[bsd$w_size] = 1;
;  166          0672  2 p[bsd$l_vbn] = 1;
;  167          0673  2 anl$bucket(p,0);
;  168          0674  2
;  169          0675  2 anl$format_skip(0);
;  170          0676  2 anl$format_skip(0);
;  171          0677  2 anl$idx_prolog(p,true,0);
```

```
;  172          0678  2
;  173          0679  2 ! Now we will check all of the area descriptors, because they describe the
;  174          0680  2 ! basic structure of the file.  Read in the first descriptor.
;  175          0681  2
;  176          0682  2 sp = .p[bsd$l_bufptr];
;  177          0683  2 area_count = .sp[plg$b_amax];
;  178          0684  2 p[bsd$l_vbn] = .sp[plg$b_avbn];
;  179          0685  2 p[bsd$l_offset] = 0;
;  180          0686  2 anl$bucket(p,0);
;  181          0687  2
;  182          0688  2 ! Now we will loop through each area descriptor.  As we go, we build up
;  183          0689  2 ! the areas vector, which tells us the bucket size for each area.
;  184          0690  2
;  185          0691  2 init_bsd(c);
;  186          0692  2 ch$fill(%x'00', 256,areas_vector);
;  187          0693  2
;  188          0694  3 incru id from 0 to .area_count-1 do (
;  189          0695  3
;  190          0696  3         ! Copy the BSD describing the area descriptor into another one, because
;  191          0697  3         ! the analysis routine will advance to the next descriptor.
;  192          0698  3
;  193          0699  3         copy_bucket(p,c);
;  194          0700  3
;  195          0701  3         ! Analyze the descriptor for validity.  This will advance the BSD on
;  196          0702  3         ! to the next one, telling us if it exists.
;  197          0703  3
;  198          0704  3         anl$format_skip(0);
;  199          0705  3         anl$area_descriptor(p,.id,true,0);
;  200          0706  3
;  201          0707  3         ! Put the bucket size into the areas vector.
;  202          0708  3
;  203          0709  3         sp = .c[bsd$l_bufptr] + .c[bsd$l_offset];
;  204          0710  3         areas_vector[.id] = .sp[area$b_arbktsz];
;  205          0711  3
;  206          0712  3         ! Now we will check any reclaimed buckets on the available list.
;  207          0713  3         ! If we are running in /SUMMARY mode, however, then user is not
;  208          0714  3         ! interested in spending the time to read through the file.
;  209          0715  3         ! Using the C BSD, get the first bucket and then loop through them.
;  210          0716  3
;  211          0717  4         if .anl$gb_mode nequ anl$k_summary and .sp[area$l_avail] nequ 0 then (
;  212          0718  4                 c[bsd$w_size] = .sp[area$b_arbktsz];
;  213          0719  4                 c[bsd$l_vbn] = .sp[area$l_avail];
;  214          0720  4                 anl$bucket(c,0);
;  215          0721  4
;  216          0722  4                 while anl$3reclaimed_bucket_header(c,false) do;
;  217          0723  3         );
;  218          0724  3
;  219          0725  3         ! If we are operating in statistics mode, we now call a routine
;  220          0726  3         ! to print the statistics that have been accumulated about this
;  221          0727  3         ! area.
;  222          0728  3
;  223          0729  3         if .anl$gb_mode eqlu anl$k_statistics then
;  224          0730  3                 anl$area_statistics(.id);
;  225          0731  2 );
```

```
;   227          0732  2 ! Now we are going to analyze the key descriptors.  Begin by setting up a
;   228          0733  2 ! BSD and reading in the first one.
;   229          0734  2
;   230          0735  2 p[bsd$l_vbn] = 1;
;   231          0736  2 p[bsd$l_offset] = 0;
;   232          0737  2 anl$bucket(p,0);
;   233          0738  2
;   234          0739  2 ! Now loop through, analyzing each one.
;   235          0740  2
;   236          0741  3 incru id from 0 do (
;   237          0742  3
;   238          0743  3         ! Copy the BSD describing this key into another one, because the
;   239          0744  3         ! analysis routine will advance to the next one.
;   240          0745  3
;   241          0746  3         copy_bucket(p,c);
;   242          0747  3
;   243          0748  3         ! Analyze the descriptor for validity.  This will advance on to the
;   244          0749  3         ! next one, telling us if there is a next one.
;   245          0750  3
;   246          0751  3         anl$format_skip(0);
;   247          0752  3         another = anl$key_descriptor(p,.id,areas_vector,true,0);
;   248          0753  3
;   249          0754  3         ! Now we want to check the complete index and data structure for
;   250          0755  3         ! this key.  We can skip it if the index is uninitialized.  Also,
;   251          0756  3         ! if we are running in /SUMMARY mode, then the user is not interested
;   252          0757  3         ! in spending the time to read through the file.
;   253          0758  3
;   254          0759  3         sp = .c[bsd$l_bufptr] + .c[bsd$l_offset];
;   255          0760  3         if .anl$gb_mode nequ anl$k_summary and not .sp[key$v_initidx] then
;   256          0761  3                 anl$idx_check_key_stuff(.sp[key$l_rootvbn],c,.sp[key$b_rootlev]);
;   257          0762  3
;   258          0763  3         ! If we are operating in statistics mode, we now call a routine
;   259          0764  3         ! to print the statistics that have been accumulated about this
;   260          0765  3         ! key.
;   261          0766  3
;   262          0767  3         if .anl$gb_mode eqlu anl$k_statistics then
;   263          0768  3                 anl$key_statistics(c);
;   264          0769  3
;   265          0770  3         ! If that was the last key descriptor, we're done.
;   266          0771  3
;   267          0772  3 exitif (not .another);
;   268          0773  2 );
;   269          0774  2
;   270          0775  2 anl$bucket(p,-1);
;   271          0776  2 anl$bucket(c,-1);
;   272          0777  2
;   273          0778  2 return;
;   274          0779  2
;   275          0780  1 end;
```

```
                                .TITLE  RMSCHECKB RMSCHECKB - Check a File Structure
                                .IDENT  \V04-000\

                                .EXTRN  ANLRMS$_OK, ANLRMS$_ALLOC
                                .EXTRN  ANLRMS$_ANYTHING
                                .EXTRN  ANLRMS$_BACKUP, ANLRMS$_BKT
```

```
.EXTRN  ANLRMS$_BKTAREA
.EXTRN  ANLRMS$_BKTCHECK
.EXTRN  ANLRMS$_BKTFLAGS
.EXTRN  ANLRMS$_BKTFREE
.EXTRN  ANLRMS$_BKTKEY, ANLRMS$_BKTLEVEL
.EXTRN  ANLRMS$_BKTNEXT
.EXTRN  ANLRMS$_BKTPTRSIZE
.EXTRN  ANLRMS$_BKTRECID
.EXTRN  ANLRMS$_BKTRECID3
.EXTRN  ANLRMS$_BKTSAMPLE
.EXTRN  ANLRMS$_BKTVBNFREE
.EXTRN  ANLRMS$_BUCKETSIZE
.EXTRN  ANLRMS$_CELL, ANLRMS$_CELLDATA
.EXTRN  ANLRMS$_CELLFLAGS
.EXTRN  ANLRMS$_CHECKHDG
.EXTRN  ANLRMS$_CONTIG, ANLRMS$_CREATION
.EXTRN  ANLRMS$_CTLSIZE
.EXTRN  ANLRMS$_DATAREC
.EXTRN  ANLRMS$_DATABKTVBN
.EXTRN  ANLRMS$_DUMPHEADING
.EXTRN  ANLRMS$_EOF, ANLRMS$_ERRORCOUNT
.EXTRN  ANLRMS$_ERRORNONE
.EXTRN  ANLRMS$_ERRORS, ANLRMS$_EXPIRATION
.EXTRN  ANLRMS$_FILEATTR
.EXTRN  ANLRMS$_FILEHDR
.EXTRN  ANLRMS$_FILEID, ANLRMS$_FILEORG
.EXTRN  ANLRMS$_FILESPEC
.EXTRN  ANLRMS$_FLAG, ANLRMS$_GLOBALBUFS
.EXTRN  ANLRMS$_HEXDATA
.EXTRN  ANLRMS$_HEXHEADING1
.EXTRN  ANLRMS$_HEXHEADING2
.EXTRN  ANLRMS$_IDXAREA
.EXTRN  ANLRMS$_IDXAREAALLOC
.EXTRN  ANLRMS$_IDXAREABKTSZ
.EXTRN  ANLRMS$_IDXAREANEXT
.EXTRN  ANLRMS$_IDXAREANOALLOC
.EXTRN  ANLRMS$_IDXAREAQTY
.EXTRN  ANLRMS$_IDXAREARECL
.EXTRN  ANLRMS$_IDXAREAUSED
.EXTRN  ANLRMS$_IDXKEY, ANLRMS$_IDXKEYAREAS
.EXTRN  ANLRMS$_IDXKEYBKTSZ
.EXTRN  ANLRMS$_IDXKEYBYTES
.EXTRN  ANLRMS$_IDXKEY1TYPE
.EXTRN  ANLRMS$_IDXKEYDATAVBN
.EXTRN  ANLRMS$_IDXKEYFILL
.EXTRN  ANLRMS$_IDXKEYFLAGS
.EXTRN  ANLRMS$_IDXKEYKEYSZ
.EXTRN  ANLRMS$_IDXKEYNAME
.EXTRN  ANLRMS$_IDXKEYNEXT
.EXTRN  ANLRMS$_IDXKEYMINREC
.EXTRN  ANLRMS$_IDXKEYNULL
.EXTRN  ANLRMS$_IDXKEYPOSS
.EXTRN  ANLRMS$_IDXKEYROOTLVL
.EXTRN  ANLRMS$_IDXKEYROOTVBN
.EXTRN  ANLRMS$_IDXKEYSEGS
.EXTRN  ANLRMS$_IDXKEYSIZES
.EXTRN  ANLRMS$_IDXPRIMREC
```

```
.EXTRN  ANLRMS$_IDXPRIMRECFLAGS
.EXTRN  ANLRMS$_IDXPRIMRECID
.EXTRN  ANLRMS$_IDXPRIMRECLEN
.EXTRN  ANLRMS$_IDXPRIMRECRRV
.EXTRN  ANLRMS$_IDXPROAREAS
.EXTRN  ANLRMS$_IDXPROLOG
.EXTRN  ANLRMS$_IDXREC, ANLRMS$_IDXRECPTR
.EXTRN  ANLRMS$_IDXSIDR
.EXTRN  ANLRMS$_IDXSIDRDUPCNT
.EXTRN  ANLRMS$_IDXSIDRFLAGS
.EXTRN  ANLRMS$_IDXSIDRRECID
.EXTRN  ANLRMS$_IDXSIDRPTRFLAGS
.EXTRN  ANLRMS$_IDXSIDRPTRREF
.EXTRN  ANLRMS$_INTERCOMMAND
.EXTRN  ANLRMS$_INTERHDG
.EXTRN  ANLRMS$_LONGREC
.EXTRN  ANLRMS$_MAXRECSIZE
.EXTRN  ANLRMS$_NOBACKUP
.EXTRN  ANLRMS$_NOEXPIRATION
.EXTRN  ANLRMS$_NOSPANFILLER
.EXTRN  ANLRMS$_PERFORM
.EXTRN  ANLRMS$_PROLOGFLAGS
.EXTRN  ANLRMS$_PROLOGVER
.EXTRN  ANLRMS$_PROT, ANLRMS$_RECATTR
.EXTRN  ANLRMS$_RECFMT, ANLRMS$_RECLAIMBKT
.EXTRN  ANLRMS$_RELBUCKET
.EXTRN  ANLRMS$_RELEOFVBN
.EXTRN  ANLRMS$_RELMAXREC
.EXTRN  ANLRMS$_RELPROLOG
.EXTRN  ANLRMS$_RELIAB, ANLRMS$_REVISION
.EXTRN  ANLRMS$_STATHDG
.EXTRN  ANLRMS$_SUMMARYHDG
.EXTRN  ANLRMS$_OWNERUIC
.EXTRN  ANLRMS$_JNL, ANLRMS$_AIJNL
.EXTRN  ANLRMS$_BIJNL, ANLRMS$_ATJNL
.EXTRN  ANLRMS$_ATTOP, ANLRMS$_BADCMD
.EXTRN  ANLRMS$_BADPATH
.EXTRN  ANLRMS$_BADVBN, ANLRMS$_DOWNHELP
.EXTRN  ANLRMS$_DOWNPATH
.EXTRN  ANLRMS$_EMPTYBKT
.EXTRN  ANLRMS$_NODATA, ANLRMS$_NODOWN
.EXTRN  ANLRMS$_NONEXT, ANLRMS$_NORECLAIMED
.EXTRN  ANLRMS$_NORECS, ANLRMS$_NORRV
.EXTRN  ANLRMS$_RESTDONE
.EXTRN  ANLRMS$_STACKFULL
.EXTRN  ANLRMS$_UNINITINDEX
.EXTRN  ANLRMS$_FDLIDENT
.EXTRN  ANLRMS$_FDLSYSTEM
.EXTRN  ANLRMS$_FDLSOURCE
.EXTRN  ANLRMS$_FDLFILE
.EXTRN  ANLRMS$_FDLALLOC
.EXTRN  ANLRMS$_FDLNOALLOC
.EXTRN  ANLRMS$_FDLBESTTRY
.EXTRN  ANLRMS$_FDLBUCKETSIZE
.EXTRN  ANLRMS$_FDLCLUSTERSIZE
.EXTRN  ANLRMS$_FDLCONTIG
.EXTRN  ANLRMS$_FDLEXTENSION
```

```
.EXTRN  ANLRMS$_FDLGLOBALBUFS
.EXTRN  ANLRMS$_FDLMAXRECORD
.EXTRN  ANLRMS$_FDLFILENAME
.EXTRN  ANLRMS$_FDLORG, ANLRMS$_FDLOWNER
.EXTRN  ANLRMS$_FDLPROTECTION
.EXTRN  ANLRMS$_FDLRECORD
.EXTRN  ANLRMS$_FDLSPAN
.EXTRN  ANLRMS$_FDLCC, ANLRMS$_FDLVFCSIZE
.EXTRN  ANLRMS$_FDLFORMAT
.EXTRN  ANLRMS$_FDLSIZE
.EXTRN  ANLRMS$_FDLAREA
.EXTRN  ANLRMS$_FDLKEY, ANLRMS$_FDLCHANGES
.EXTRN  ANLRMS$_FDLDATAAREA
.EXTRN  ANLRMS$_FDLDATAFILL
.EXTRN  ANLRMS$_FDLDATAKEYCOMPB
.EXTRN  ANLRMS$_FDLDATARECCOMPB
.EXTRN  ANLRMS$_FDLDUPS
.EXTRN  ANLRMS$_FDLINDEXAREA
.EXTRN  ANLRMS$_FDLINDEXCOMPB
.EXTRN  ANLRMS$_FDLINDEXFILL
.EXTRN  ANLRMS$_FDLL1INDEXAREA
.EXTRN  ANLRMS$_FDLKEYNAME
.EXTRN  ANLRMS$_FDLNORECS
.EXTRN  ANLRMS$_FDLNULLKEY
.EXTRN  ANLRMS$_FDLNULLVALUE
.EXTRN  ANLRMS$_FDLPROLOG
.EXTRN  ANLRMS$_FDLSEGLENGTH
.EXTRN  ANLRMS$_FDLSEGPOS
.EXTRN  ANLRMS$_FDLSEGTYPE
.EXTRN  ANLRMS$_FDLANALAREA
.EXTRN  ANLRMS$_FDLRECL
.EXTRN  ANLRMS$_FDLANALKEY
.EXTRN  ANLRMS$_FDLDATAKEYCOMP
.EXTRN  ANLRMS$_FDLDATARECCOMP
.EXTRN  ANLRMS$_FDLDATARECS
.EXTRN  ANLRMS$_FDLDATASPACE
.EXTRN  ANLRMS$_FDLDEPTH
.EXTRN  ANLRMS$_FDLDUPSPER
.EXTRN  ANLRMS$_FDLIDXCOMP
.EXTRN  ANLRMS$_FDLIDXFILL
.EXTRN  ANLRMS$_FDLIDXSPACE
.EXTRN  ANLRMS$_FDLIDXL1RECS
.EXTRN  ANLRMS$_FDLDATALENMEAN
.EXTRN  ANLRMS$_FDLIDXLENMEAN
.EXTRN  ANLRMS$_STATAREA
.EXTRN  ANLRMS$_STATRECL
.EXTRN  ANLRMS$_STATKEY
.EXTRN  ANLRMS$_STATDEPTH
.EXTRN  ANLRMS$_STATIDXL1RECS
.EXTRN  ANLRMS$_STATIDXLENMEAN
.EXTRN  ANLRMS$_STATIDXSPACE
.EXTRN  ANLRMS$_STATIDXFILL
.EXTRN  ANLRMS$_STATIDXCOMP
.EXTRN  ANLRMS$_STATDATARECS
.EXTRN  ANLRMS$_STATDUPSPER
.EXTRN  ANLRMS$_STATDATALENMEAN
.EXTRN  ANLRMS$_STATDATASPACE
```

```
.EXTRN  ANLRMS$_STATDATAFILL
.EXTRN  ANLRMS$_STATDATAKEYCOMP
.EXTRN  ANLRMS$_STATDATARECCOMP
.EXTRN  ANLRMS$_STATEFFICIENCY
.EXTRN  ANLRMS$_BADAREA1ST2
.EXTRN  ANLRMS$_BADAREABKTSIZE
.EXTRN  ANLRMS$_BADAREAFIT
.EXTRN  ANLRMS$_BADAREAID
.EXTRN  ANLRMS$_BADAREANEXT
.EXTRN  ANLRMS$_BADAREAROOT
.EXTRN  ANLRMS$_BADAREAUSED
.EXTRN  ANLRMS$_BADBKTAREAID
.EXTRN  ANLRMS$_BADBKTCHECK
.EXTRN  ANLRMS$_BADBKTFREE
.EXTRN  ANLRMS$_BADBKTKEYID
.EXTRN  ANLRMS$_BADBKTLEVEL
.EXTRN  ANLRMS$_BADBKTROOTBIT
.EXTRN  ANLRMS$_BADBKTSAMPLE
.EXTRN  ANLRMS$_BADCELLFIT
.EXTRN  ANLRMS$_BADCHECKSUM
.EXTRN  ANLRMS$_BADDATARECBITS
.EXTRN  ANLRMS$_BADDATARECFIT
.EXTRN  ANLRMS$_BADDATARECPS
.EXTRN  ANLRMS$_BAD3IDXKEYFIT
.EXTRN  ANLRMS$_BADIDXLASTKEY
.EXTRN  ANLRMS$_BADIDXORDER
.EXTRN  ANLRMS$_BADIDXRECBITS
.EXTRN  ANLRMS$_BADIDXRECFIT
.EXTRN  ANLRMS$_BADIDXRECPS
.EXTRN  ANLRMS$_BADKEYAREAID
.EXTRN  ANLRMS$_BADKEYDATABKT
.EXTRN  ANLRMS$_BADKEYDATAFIT
.EXTRN  ANLRMS$_BADKEYDATATYPE
.EXTRN  ANLRMS$_BADKEYIDXBKT
.EXTRN  ANLRMS$_BADKEYFILL
.EXTRN  ANLRMS$_BADKEYFIT
.EXTRN  ANLRMS$_BADKEYREFID
.EXTRN  ANLRMS$_BADKEYROOTLEVEL
.EXTRN  ANLRMS$_BADKEYSEGCOUNT
.EXTRN  ANLRMS$_BADKEYSEGVEC
.EXTRN  ANLRMS$_BADKEYSUMMARY
.EXTRN  ANLRMS$_BADREADNOPAR
.EXTRN  ANLRMS$_BADREADPAR
.EXTRN  ANLRMS$_BADSIDRDUPCT
.EXTRN  ANLRMS$_BADSIDRPTRFIT
.EXTRN  ANLRMS$_BADSIDRPTRSZ
.EXTRN  ANLRMS$_BADSIDRSIZE
.EXTRN  ANLRMS$_BADSTREAMEOF
.EXTRN  ANLRMS$_BADVBNFREE
.EXTRN  ANLRMS$_BKTLOOP
.EXTRN  ANLRMS$_EXTENDERR
.EXTRN  ANLRMS$_FLAGERROR
.EXTRN  ANLRMS$_MISSINGBKT
.EXTRN  ANLRMS$_NOTOK, ANLRMS$_SPANERROR
.EXTRN  ANLRMS$_TOOMANYRECS
.EXTRN  ANLRMS$_UNWIND, ANLRMS$_VFCTOOSHORT
.EXTRN  ANLRMS$_CACHEFULL
```

```
                                                        .EXTRN  ANLRMS$_CACHERELFAIL
                                                        .EXTRN  ANLRMS$_FACILITY
                                                        .EXTRN  ANL$AREA_DESCRIPTOR
                                                        .EXTRN  ANL$AREA_STATISTICS
                                                        .EXTRN  ANL$BUCKET, ANL$2BUCKET_HEADER
                                                        .EXTRN  ANL$3BUCKET_HEADER                              ;
                                                        .EXTRN  ANL$FORMAT_ERROR
                                                        .EXTRN  ANL$FORMAT_LINE
                                                        .EXTRN  ANL$FORMAT_SKIP                                 ;
                                                        .EXTRN  ANL$IDX_PROLOG, ANL$2INDEX_RECORD               ;
                                                        .EXTRN  ANL$3INDEX_RECORD
                                                        .EXTRN  ANL$KEY_DESCRIPTOR
                                                        .EXTRN  ANL$KEY_STATISTICS
                                                        .EXTRN  ANL$UNWIND_HANDLER
                                                        .EXTRN  ANL$2PRIMARY_DATA_RECORD
                                                        .EXTRN  ANL$3PRIMARY_DATA_RECORD
                                                        .EXTRN  ANL$PROLOG_CHECKSUMS                            ;
                                                        .EXTRN  ANL$3RECLAIMED_BUCKET_HEADER                    ;
                                                        .EXTRN  ANL$2SIDR_RECORD                                ;
                                                        .EXTRN  ANL$3SIDR_RECORD                                ;
                                                        .EXTRN  LIB$ESTABLISH, ANL$GB_MODE                      ;
                                                        .EXTRN  ANL$GL_FAT, ANL$GW_PROLOG                       ;

                                                        .PSECT  $CODE$,NOWRT,2

                                        07FC 00000      .ENTRY  ANL$IDX_CHECK, Save R2,R3,R4,R5,R6,R7,R8,-      ; 0648
                                                                R9,R10                                          ;
                        5A      0000G   CF  9E 00002    MOVAB   ANL$GB_MODE, R10                                ;
                        59      0000G   CF  9E 00007    MOVAB   ANL$FORMAT_SKIP, R9                             ;
                        58      0000G   CF  9E 0000C    MOVAB   ANL$BUCKET, R8                                  ;
                        5E      FED0    CE  9E 00011    MOVAB   -304(SP), SP                                    ;
                                0000G   CF  9F 00016    PUSHAB  ANL$UNWIND_HANDLER                              ; 0661
                    00000000G   00          01  FB 0001A CALLS   #1, LIB$ESTABLISH                           ;
                        0000G   CF          00  FB 00021 CALLS   #0, ANL$PROLOG_CHECKSUMS                    ; 0665
18              00              6E          00  2C 00026 MOVC5   #0, (SP), #0, #24, P                       ; 0670
                                        E8  AD      0002B                                                   ;
                            EA  AD          01  B0 0002D MOVW    #1, P+2                                    ; 0671
                            EC  AD          01  D0 00031 MOVL    #1, P+4                                    ; 0672
                                            7E  D4 00035 CLRL    -(SP)                                      ; 0673
                                        E8  AD  9F 00037 PUSHAB  P                                          ;
                                68          02  FB 0003A CALLS   #2, ANL$BUCKET                             ;
                                            7E  D4 0003D CLRL    -(SP)                                      ; 0675
                                69          01  FB 0003F CALLS   #1, ANL$FORMAT_SKIP                         ;
                                            7E  D4 00042 CLRL    -(SP)                                      ; 0676
                                69          01  FB 00044 CALLS   #1, ANL$FORMAT_SKIP                         ;
                                7E          01  7D 00047 MOVQ    #1, -(SP)                                  ; 0677
                                        E8  AD  9F 0004A PUSHAB  P                                          ;
                            0000G   CF          03  FB 0004D CALLS   #3, ANL$IDX_PROLOG                     ;
                                57      F4  AD  D0 00052 MOVL    P+12, SP                                   ; 0682
                                56      67  A7  9A 00056 MOVZBL  103(SP), AREA_COUNT                        ; 0683
                            EC  AD      66  A7  9A 0005A MOVZBL  102(SP), P+4                               ; 0684
                                        F0  AD  D4 0005F CLRL    P+8                                        ; 0685
                                            7E  D4 00062 CLRL    -(SP)                                      ; 0686
                                        E8  AD  9F 00064 PUSHAB  P                                          ;
                                68          02  FB 00067 CALLS   #2, ANL$BUCKET                             ;
18              00              6E          00  2C 0006A MOVC5   #0, (SP), #0, #24, C                       ; 0691
                                        D0  AD      0006F                                                   ;
```

```
0100   8F          00          6E          00  2C 00071        MOVC5   #0, (SP), #0, #256, AREAS_VECTOR          ; 0692
                                           6E     00078                                                       ;
                                           56  D7 00079        DECL    R6                                     ; 0694
                                           52  D4 0007B        CLRL    ID                                     ;
                                           6B  11 0007D        BRB     5$                                     ;
                         D0    AD     E8   AD  7D 0007F 1$:    MOVQ    F, T                                   ; 0699
                         D8    AD     F0   AD  D0 00084        MOVL    F+8, T+8                               ;
                         E4    AD     FC   AD  D0 00089        MOVL    F+20, T+20                             ;
                                           7E  D4 0008E        CLRL    -(SP)                                  ;
                                      D0   AD  9F 00090        PUSHAB  T                                      ;
                               68          02  FB 00093        CALLS   #2, ANL$BUCKET                         ;
                                           7E  D4 00096        CLRL    -(SP)                                  ; 0704
                               69          01  FB 00098        CALLS   #1, ANL$FORMAT_SKIP                    ;
                               7E          01  7D 0009B        MOVQ    #1, -(SP)                              ; 0705
                                           52  DD 0009E        PUSHL   ID                                     ;
                                      E8   AD  9F 000A0        PUSHAB  P                                      ;
                      0000G    CF          04  FB 000A3        CALLS   #4, ANL$AREA_DESCRIPTOR                ;
              57         DC    AD     D8   AD  C1 000A8        ADDL3   C+8, C+12, SP                          ; 0709
                             6E42     03   A7  90 000AE        MOVB    3(SP), AREAS_VECTOR[ID]                ; 0710
                               05          6A  91 000B3        CMPB    ANL$GB_MODE, #5                        ; 0717
                                           24  13 000B6        BEQL    3$                                     ;
                                      08   A7  D5 000B8        TSTL    8(SP                                   ;
                                           1F  13 000BB        BEQL    3$                                     ;
                         D2    AD     03   A7  9B 000BD        MOVZBW  3(SP), C+2                             ; 0718
                         D4    AD     08   A7  D0 000C2        MOVL    8(SP), C+4                             ; 0719
                                           7E  D4 000C7        CLRL    -(SP)                                  ; 0720
                                      D0   AD  9F 000C9        PUSHAB  C                                      ;
                               68          02  FB 000CC        CALLS   #2, ANL$BUCKET                         ;
                                           7E  D4 000CF 2$:    CLRL    -(SP)                                  ; 0722
                                      D0   AD  9F 000D1        PUSHAB  C                                      ;
                      0000G    CF          02  FB 000D4        CALLS   #2, ANL$3RECLAIMED_BUCKET_HEADER       ;
                               F3          50  E8 000D9        BLBS    R0, 2$                                 ;
                               04          6A  91 000DC 3$:    CMPB    ANL$GB_MODE, #4                        ; 0729
                                           07  12 000DF        BNEQ    4$                                     ;
                                           52  DD 000E1        PUSHL   ID                                     ; 0730
                      0000G    CF          01  FB 000E3        CALLS   #1, ANL$AREA_STATISTICS                ;
                                           52  D6 000E8 4$:    INCL    ID                                     ; 0694
                               56          52  D1 000EA 5$:    CMPL    ID, R6                                 ;
                                           90  1B 000ED        BLEQU   1$                                     ;
                         EC    AD          01  7D 000EF        MOVQ    #1, P+4                                ; 0735
                                           7E  D4 000F3        CLRL    -(SP)                                  ; 0737
                                      E8   AD  9F 000F5        PUSHAB  P                                      ;
                               68          02  FB 000F8        CALLS   #2, ANL$BUCKET                         ;
                                           52  D4 000FB        CLRL    ID                                     ; 0741
                         D0    AD     E8   AD  7D 000FD 6$:    MOVQ    F, T                                   ; 0746
                         D8    AD     F0   AD  D0 00102        MOVL    F+8, T+8                               ;
                         E4    AD     FC   AD  D0 00107        MOVL    F+20, T+20                             ;
                                           7E  D4 0010C        CLRL    -(SP)                                  ;
                                      D0   AD  9F 0010E        PUSHAB  T                                      ;
                               68          02  FB 00111        CALLS   #2, ANL$BUCKET                         ;
                                           7E  D4 00114        CLRL    -(SP)                                  ; 0751
                               69          01  FB 00116        CALLS   #1, ANL$FORMAT_SKIP                    ;
                               7E          01  7D 00119        MOVQ    #1, -(SP)                              ; 0752
                                      08   AE  9F 0011C        PUSHAB  AREAS_VECTOR                           ;
                                           52  DD 0011F        PUSHL   ID                                     ;
                                      E8   AD  9F 00121        PUSHAB  P                                      ;
                      0000G    CF          05  FB 00124        CALLS   #5, ANL$KEY_DESCRIPTOR                 ;
```

```
                       53             50  90 00129          MOVB    R0, ANOTHER                              ;
         57      DC    AD      D8     AD  C1 0012C          ADDL3   C+8, C+12, SP                            ; 0759
                       05             6A  91 00132          CMPB    ANL$GB_MODE, #5                          ; 0760
                                      14  13 00135          BEQL    7$                                       ;
         0F      10    A7             04  E0 00137          BBS     #4, 16(SP), 7$                           ;
                       7E      09     A7  9A 0013C          MOVZBL  9(SP), -(SP)                             ; 0761
                               D0     AD  9F 00140          PUSHAB  C                                        ;
                               0C     A7  DD 00143          PUSHL   12(SP)                                   ;
                0000V  CF             03  FB 00146          CALLS   #3, ANL$IDX_CHECK_KEY_STUFF              ;
                       04             6A  91 0014B 7$:      CMPB    ANL$GB_MODE, #4                          ; 0767
                                      08  12 0014E          BNEQ    8$                                       ;
                               D0     AD  9F 00150          PUSHAB  C                                        ; 0768
                0000G  CF             01  FB 00153          CALLS   #1, ANL$KEY_STATISTICS                   ;
                       04             53  E9 00158 8$:      BLBC    ANOTHER, 9$                              ; 0772
                                      52  D6 0015B          INCL    ID                                       ; 0741
                                      9E  11 0015D          BRB     6$                                       ;
                       7E             01  CE 0015F 9$:      MNEGL   #1, -(SP)                                ; 0775
                               E8     AD  9F 00162          PUSHAB  P                                        ;
                       68             02  FB 00165          CALLS   #2, ANL$BUCKET                           ;
                       7E             01  CE 00168          MNEGL   #1, -(SP)                                ; 0776
                               D0     AD  9F 0016B          PUSHAB  C                                        ;
                       68             02  FB 0016E          CALLS   #2, ANL$BUCKET                           ;
                                          04 00171          RET                                              ; 0780

; Routine Size:  370 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  277          0781  1 %sbttl 'ANL$IDX_CHECK_KEY_STUFF - Check Structure of Index & Data'
;  278          0782  1 !++
;  279          0783  1 ! Functional Description:
;  280          0784  1 !       This routine is called to check the structure of the index and
;  281          0785  1 !       data buckets for a key.  It scans the index entries and data
;  282          0786  1 !       records in order, checking for structural flaws.
;  283          0787  1 !
;  284          0788  1 !       It is a requirement of this routine that it visit each bucket
;  285          0789  1 !       exactly once.  This is because the routine is also used to collect
;  286          0790  1 !       statistics about the key.
;  287          0791  1 !
;  288          0792  1 ! Formal Parameters:
;  289          0793  1 !       vbn             The VBN of the index bucket to be checked.  On first
;  290          0794  1 !                       call, this is the root bucket.  On recursions, it is
;  291          0795  1 !                       some lower-level index bucket.
;  292          0796  1 !       key_bsd         Address of BSD for the key descriptor.
;  293          0797  1 !       level           The alleged level of this index bucket.
;  294          0798  1 !
;  295          0799  1 ! Implicit Inputs:
;  296          0800  1 !       global data
;  297          0801  1 !
;  298          0802  1 ! Implicit Outputs:
;  299          0803  1 !       global data
;  300          0804  1 !
;  301          0805  1 ! Returned Value:
;  302          0806  1 !       none
;  303          0807  1 !
;  304          0808  1 ! Side Effects:
;  305          0809  1 !
;  306          0810  1 !--
;  307          0811  1
;  308          0812  1
;  309          0813  2 global routine anl$idx_check_key_stuff(vbn,key_bsd,level): novalue = begin
;  310          0814  2
;  311          0815  2 bind
;  312          0816  2         prolog_3 = .anl$gw_prolog eqlu plg$c_ver_3,
;  313          0817  2         k = .key_bsd: bsd;
;  314          0818  2
;  315          0819  2 own
;  316          0820  2         bucket_count: signed long,
;  317          0821  2         d: bsd;
;  318          0822  2
;  319          0823  2 local
;  320          0824  2         kp: ref block[,byte],
;  321          0825  2         hp: ref block[,byte],
;  322          0826  2         rp: ref block[,byte],
;  323          0827  2         vp: ref block[,byte],
;  324          0828  2         b: bsd,
;  325          0829  2         another_record: byte,
;  326          0830  2         down_vbn: long;
;  327          0831  2
;  328          0832  2
;  329          0833  2 ! We will be referencing the key descriptor throughout this routine.
;  330          0834  2
;  331          0835  2 kp = .k[bsd$l_bufptr] + .k[bsd$l_offset];
;  332          0836  2
;  333          0837  2 ! We have to do some initialization based upon whether this is the root
```

```
;  334          0838  2 ! bucket or not (i.e., on the first call).
;  335          0839  2
;  336          0840  2 if .level eqlu .kp[key$b_rootlev] then (
;  337          0841  3
;  338          0842  3         ! We want to detect loops in the bucket structure.  To do this,
;  339          0843  3         ! we will initialize a counter to the maximum possible number
;  340          0844  3         ! of buckets that can appear for this key.  If the count ever
;  341          0845  3         ! goes to zero as we read a bucket, we're in trouble.
;  342          0846  3
;  343          0847  3         bucket_count =          .anl$gl_fat[fat$l_hiblk] /
;  344          0848  3                         minu(.kp[key$b_idxbktsz],.kp[key$b_datbktsz]);
;  345          0849  3
;  346          0850  3         ! We will be scanning all of the data buckets for this key when
;  347          0851  3         ! we get down to the lowest-level index buckets.  Let's build a
;  348          0852  3         ! BSD for scanning these buckets now, and leave it around during
;  349          0853  3         ! all recursive calls.
;  350          0854  3
;  351          0855  3         init_bsd(d);
;  352          0856  3         d[bsd$w_size] = .kp[key$b_datbktsz];
;  353          0857  3         d[bsd$l_vbn] = .kp[key$l_[dvbn];
;  354          0858  3         anl$bucket(d,.k[bsd$l_vbn]);
;  355          0859  3
;  356          0860  2 );
```

```
;  358          0861  2 ! Begin by setting up a BSD for the index bucket we are to check.  Read in
;  359          0862  2 ! the bucket and set up a pointer to the header.
;  360          0863  2
;  361          0864  2 init_bsd(b);
;  362          0865  2 b[bsd$w_size] = .kp[key$b_idxbktsz];
;  363          0866  2 b[bsd$l_vbn] = .vbn;
;  364          0867  2 anl$bucket(b,0);
;  365          0868  2 hp = .b[bsd$l_bufptr];
;  366          0869  2
;  367          0870  2 ! Decrement the count of possible buckets.  If it goes to zero, then
;  368          0871  2 ! there is a loop in the index structure.
;  369          0872  2
;  370          0873  3 if decrement(bucket_count) lss 0 then (
;  371          0874  3         anl$format_error(anlrms$_bktloop,.vbn,.kp[key$b_keyref]);
;  372          0875  3         signal (anlrms$_unwind);
;  373          0876  2 );
;  374          0877  2
;  375          0878  2 ! Now we want to check the integrity of the bucket header.  The routine
;  376          0879  2 ! we call will update the BSD to describe the next bucket in the chain.
;  377          0880  2 ! We don't want this, so we use a copy of the BSD.
;  378          0881  2
;  379          0882  3 begin
;  380          0883  3 local
;  381          0884  3         h: bsd;
;  382          0885  3
;  383          0886  3 init_bsd(h);
;  384          0887  3 copy_bucket(b,h);
;  385          0888  3 if prolog_3 then
;  386          0889  3         anl$3bucket_header(h,.kp[key$b_keyref],.kp[key$v_dupkeys],.level,false)
;  387          0890  3 else
;  388          0891  4         anl$2bucket_header(h,(if .level eqlu 1 then .kp[key$b_lanum]
;  389          0892  3                                                else .kp[key$b_ianum]),.level,false);
;  390          0893  3 anl$bucket(h,-1);
;  391          0894  2 end;
;  392          0895  2
;  393          0896  2 ! Now we can check the root bucket bit in the header to make sure it is
;  394          0897  2 ! correct.  If not, we better just forget it.
;  395          0898  2
;  396          0899  3 if .level eqlu .kp[key$b_rootlev] xor .hp[bkt$v_rootbkt] then (
;  397          0900  3         anl$format_error(anlrms$_badbktrootbit,.b[bsd$l_vbn]);
;  398          0901  3         signal (anlrms$_unwind);
;  399          0902  2 );
;  400          0903  2
```

```
;  402          0904  2 ! We are ready to scan the index records in this bucket.  Set up the
;  403          0905  2 ! BSD to point at the first one.  The work longword will count them as
;  404          0906  2 ! we go.
;  405          0907  2
;  406          0908  2 b[bsd$l_offset] = bkt$c_overhdsz;
;  407          0909  2 b[bsd$l_work] = 0;
;  408          0910  2
;  409          0911  3 do (
;  410          0912  3
;  411          0913  3         ! Save a pointer to the index record to be checked on this iteration.
;  412          0914  3         ! In the case of prolog 3, we also need a pointer to the VBN in the
;  413          0915  3         ! VBN list at the end of the bucket.
;  414          0916  3
;  415          0917  3         rp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
;  416          0918  3         if prolog_3 then
;  417          0919  3                 vp = (.b[bsd$l_endptr]-4) - (.b[bsd$l_work]+1) * (.hp[bkt$v_ptr_sz]+2);
;  418          0920  3
;  419          0921  3         ! Now we call a routine to analyze the index record, which will
;  420          0922  3         ! cause the BSD to be updated to the next record.  A flag is returned
;  421          0923  3         ! to tell us if there is another record.
;  422          0924  3
;  423          0925  4         another_record = (if prolog_3 then anl$3index_record
;  424          0926  3                                       else anl$2index_record) (b,k,false);
;  425          0927  3
;  426          0928  3         ! We need to extract the bucket pointer from this index record.
;  427          0929  3         ! We also want RP to point at the actual key, which is already
;  428          0930  3         ! the case for prolog 3.
;  429          0931  3
;  430          0932  3         if prolog_3 then
;  431          0933  4                 down_vbn =      (case .hp[bkt$v_ptr_sz] from 0 to 2 of set
;  432          0934  4                                 [0]:    .vp[0,0,16,0];
;  433          0935  4                                 [1]:    .vp[0,0,24,0];
;  434          0936  4                                 [2]:    .vp[0,0,32,0];
;  435          0937  4                                 tes)
;  436          0938  4
;  437          0939  4         else (
;  438          0940  5                 down_vbn =      (case .rp[irc$v_ptrsz] from 0 to 2 of set
;  439          0941  5                                 [0]:    .rp[1,0,16,0];
;  440          0942  5                                 [1]:    .rp[1,0,24,0];
;  441          0943  5                                 [2]:    .rp[1,0,32,0];
;  442          0944  4                                 tes);
;  443          0945  4                 rp = .rp + .rp[irc$v_ptrsz] + 3;
;  444          0946  3         );
```

```
;  446          0947  3          ! Now we want to analyze the bucket that the bucket pointer
;  447          0948  3          ! referenced.
;  448          0949  3
;  449          0950  3          if .level gequ 2 then
;  450          0951  3
;  451          0952  3                  ! This index entry references a deeper index bucket.
;  452          0953  3                  ! Recurse to analyze the new index bucket.
;  453          0954  3
;  454          0955  3                  anl$idx_check_key_stuff(.down_vbn,k,.level-1)
;  455          0956  3
;  456          0957  4          else (
;  457          0958  4
;  458          0959  4                  ! It references a data bucket.  We want to read the data
;  459          0960  4                  ! bucket and check it.  However, it is possible that some
;  460          0961  4                  ! data buckets exist between the last one we checked
;  461          0962  4                  ! and this new one (for example, if a bunch of SIDR duplicates
;  462          0963  4                  ! take more than one bucket to store).  So let's read and
;  463          0964  4                  ! check all buckets up to and including this new one.
;  464          0965  4
;  465          0966  4                  bind
;  466          0967  4                          highest_key = .level eqlu 1 and
;  467          0968  4                                        .hp[bkt$v_lastbkt] and
;  468          0969  4                                        not .another_record       : byte;
;  469          0970  4                  local
;  470          0971  4                          r: bsd,
;  471          0972  4                          another_bucket: byte;
;  472          0973  4
;  473          0974  4
;  474          0975  4                  ! Now we go into a loop, once through for each bucket we
;  475          0976  4                  ! want to check.
;  476          0977  4
;  477          0978  4                  init_bsd(r);
;  478          0979  4
;  479          0980  5                  loop (
;  480          0981  5
;  481          0982  5                          local
;  482          0983  5                                  hp: ref block[,byte];
;  483          0984  5
;  484          0985  5                          ! We want to check the header of the bucket.  This
;  485          0986  5                          ! will update the BSD to describe the next bucket.
;  486          0987  5                          ! Make a copy of the BSD before so we can get at
;  487          0988  5                          ! the bucket contents below.
;  488          0989  5
;  489          0990  5                          copy_bucket(d,r);
;  490          0991  6                          another_bucket = (if prolog_3 then
;  491          0992  6                                                  anl$3bucket_header(d,.kp[key$b_keyref],
;  492          0993  6                                                                     .kp[key$v_dupkeys],0,false)
;  493          0994  6                                            else
;  494          0995  5                                                  anl$2bucket_header(d,.kp[key$b_danum],0,false));
;  495          0996  5
;  496          0997  5                          ! Now we want to loop through the data records in the
;  497          0998  5                          ! bucket, if there are any.  Set up the BSD for the
;  498          0999  5                          ! first one.  Then loop for each record, allowing the
;  499          1000  5                          ! analysis routine to update the BSD.
;  500          1001  5
;  501          1002  5                          hp = .r[bsd$l_bufptr];
;  502          1003  6                          if .hp[bkt$w_freespace] gtru bkt$c_overhdsz then (
```

```
;  503          1004  6                                    r[bsd$l_offset] = bkt$c_overhdsz;
;  504          1005  6
;  505          1006  8                                    while ((if prolog_3 then
;  506          1007  8                                                if .kp[key$b_keyref] eqlu 0 then
;  507          1008  8                                                    anl$3primary_data_record
;  508          1009  8                                                else
;  509          1010  8                                                    anl$3sidr_record
;  510          1011  8                                            else
;  511          1012  8                                                if .kp[key$b_keyref] eqlu 0 then
;  512          1013  8                                                    anl$2primary_data_record
;  513          1014  8                                                else
;  514          1015  6                                                    anl$2sidr_record) (r,k,false)) do;
;  515          1016  5                                );
;  516          1017  5
;  517          1018  5                                ! Decrement the count of possible buckets.  If it
;  518          1019  5                                ! goes to zero, then there is a loop in the data
;  519          1020  5                                ! bucket structure.
;  520          1021  5
;  521          1022  6                                if decrement(bucket_count) lss 0 then (
;  522          1023  6                                    anl$format_error(anlrms$_bktloop,,r[bsd$l_vbn],,kp[key$b_keyref]);
;  523          1024  6                                    signal (anlrms$_unwind);
;  524          1025  5                                );
;  525          1026  5
;  526          1027  5                                ! The following absurdity determines when we are
;  527          1028  5                                ! done checking data buckets on this iteration.
;  528          1029  5                                ! If we're at the highest key in the index, we check
;  529          1030  5                                ! all remaining buckets.  If not, then we check
;  530          1031  5                                ! buckets until we "catch up" to the index entry.
;  531          1032  5
;  532          1033  6                                if highest_key then (
;  533          1034  6                                    exitif(not .another_bucket);
;  534          1035  5                                ) else
;  535          1036  6                                    if .r[bsd$l_vbn] eqlu .down_vbn then (
;  536          1037  6                                        exitloop;
;  537          1038  5                                    ) else
;  538          1039  6                                        if not .another_bucket then (
;  539          1040  6                                            anl$format_error(anlrms$_missingbkt,,b[bsd$l_vbn],,down_vbn)
;  540          1041  6                                            signal (anlrms$_unwind);
;  541          1042  5                                        );
;  542          1043  5
;  543          1044  4                                );
;  544          1045  4
;  545          1046  4                                anl$bucket(r,-1);
;  546          1047  3                        );
;  547          1048  3
;  548          1049  3                        ! Continue on to the next index record.
;  549          1050  3
;  550          1051  2 ) while .another_record;
```

```
;   2 ! Free up bucket buffers.
;   2
;   2 anl$bucket(b,-1);
;   2 if .level eqlu .kp[key$b_rootlev] then
;   2         anl$bucket(d,-1);
;   2
;   2 return;
;   2
;   1 end;


                                                                          .PSECT  $OWN$,NOEXE,2

                                                    00000 BUCKET_COUNT:
                                                                          .BLKB   4
                                                    00004 D:              .BLKB   24

                                                          F=                      D

                                                                          .PSECT  $CODE$,NOWRT,2

                                                OFFC 00000                .ENTRY  ANL$IDX_CHECK_KEY_STUFF, Save R2,R3,R4,R5,- ; 0813
                                                                                  R6,R7,R8,R9,R10,R11                         ;
                             5E        C0    AE  9E 00002                 MOVAB   -64(SP), SP                                 ;
                                       04    AE  D4 00006                 CLRL    4(SP)                                       ; 0816
                             03     0000G    CF  B1 00009                 CMPW    ANL$GW_PROLOG, #3                           ;
                                             03  12 0000E                 BNEQ    1$                                          ;
                                       04    AE  D6 00010                 INCL    4(SP)                                       ;
                             5B        08    AC  D0 00013 1$:             MOVL    KEY_BSD, R11                                ; 0817
                  56         0C    AB  08    AB  C1 00017                 ADDL3   8(R11), 12(R11), KP                         ; 0835
                                             6E  D4 0001D                 CLRL    (SP)                                        ; 0840
   0C    AC      09    A6          08        00  ED 0001F                 CMPZV   #0, #8, 9(KP), LEVEL                        ;
                                             3C  12 00026                 BNEQ    3$                                          ;
                                             6E  D6 00028                 INCL    (SP)                                        ;
                             51     0000G    CF  D0 0002A                 MOVL    ANL$GL_FAT, R1                              ; 0847
                             50        0A    A6  9A 0002F                 MOVZBL  10(KP), R0                                  ; 0848
                             50        0B    A6  91 00033                 CMPB    11(KP), R0                                  ;
                                             04  1E 00037                 BGEQU   2$                                          ;
                             50        0B    A6  9A 00039                 MOVZBL  11(KP), R0                                  ;
                0000'   CF         04    A1  50  C7 0003D 2$:             DIVL3   R0, 4(R1), BUCKET_COUNT                     ;
        18              00         6E        00  2C 00044                 MOVC5   #0, (SP), #0, #24, D                        ; 0855
                                    0000'    CF     00049                                                                 ;
                0000'   CF         0B    A6  9B 0004C                 MOVZBW  11(KP), D+2                                 ; 0856
                0000'   CF         54    A6  D0 00052                 MOVL    84(KP), D+4                                 ; 0857
                                       04    AB  DD 00058                 PUSHL   4(R11)                                      ; 0858
                                    0000'    CF  9F 0005B                 PUSHAB  D                                           ;
                0000G   CF                   02  FB 0005F                 CALLS   #2, ANL$BUCKET                              ;
        18              00         6E        00  2C 00064 3$:             MOVC5   #0, (SP), #0, #24, B                        ; 0864
                                       28    AE     00069                                                                 ;
                  2A    AE         0A    A6  9B 0006B                 MOVZBW  10(KP), B+2                                 ; 0865
                  2C    AE         04    AC  D0 00070                 MOVL    VBN, B+4                                    ; 0866
                                             7E  D4 00075                 CLRL    -(SP)                                       ; 0867
                                       2C    AE  9F 00077                 PUSHAB  B                                           ;
                0000G   CF                   02  FB 0007A                 CALLS   #2, ANL$BUCKET                              ;
                             59        34    AE  D0 0007F                 MOVL    B+12, HP                                    ; 0868
```

```
                                 1F      0000'  CF  F4 00083          SOBGEQ  BUCKET_COUNT, 4$                          ; 0873
                                 7E         15  A6  9A 00088          MOVZBL  21(KP), -(SP)                             ; 0874
                                            04  AC  DD 0008C          PUSHL   VBN                                       ;
                                 00000000G  8F  DD 0008F             PUSHL   #ANLRMS$_BKTLOOP                          ;
                          0000G  CF             03  FB 00095          CALLS   #3, ANL$FORMAT_ERROR                      ;
                                 00000000G  8F  DD 0009A             PUSHL   #ANLRMS$_UNWIND                           ; 0875
                    00000000G  00               01  FB 000A0          CALLS   #1, LIB$SIGNAL                            ;
18            00                6E               00  2C 000A7 4$:     MOVC5   #0, (SP), #0, #24, H                      ; 0886
                                            10  AE                    000AC                                             ;
                             10   AE         28  AE  7D 000AE          MOVQ    F, T                                      ; 0887
                             18   AE         30  AE  D0 000B3          MOVL    F+8, T+8                                  ;
                             24   AE         3C  AE  DC 000B8          MOVL    F+20, T+20                                ;
                                             7E  D4  000BD             CLRL    -(SP)                                     ;
                                            14   AE  9F 000BF          PUSHAB  T                                         ;
                          0000G  CF             02  FB 000C2          CALLS   #2, ANL$BUCKET                            ;
                                 19          04  AE  E9 000C7          BLBC    4(SP), 5$                                 ; 0888
                                             7E  D4  000CB             CLRL    -(SP)                                     ; 0889
                                            0C   AC  DD 000CD          PUSHL   LEVEL                                     ;
7E            10   A6                  01       00  EF 000D0          EXTZV   #0, #1, 16(KP), -(SP)                     ;
                                 7E         15  A6  9A 000D6          MOVZBL  21(KP), -(SP)                             ;
                                            20   AE  9F 000DA          PUSHAB  H                                         ;
                          0000G  CF             05  FB 000DD          CALLS   #5, ANL$3BUCKET_HEADER                    ;
                                             1D  11  000E2             BRB     8$                                        ;
                                             7E  D4  000E4 5$:         CLRL    -(SP)                                     ; 0891
                                            0C   AC  DD 000E6          PUSHL   LEVEL                                     ; 0892
                                 01         0C   AC  D1 000E9          CMPL    LEVEL, #1                                 ; 0891
                                             06  12  000ED             BNEQ    6$                                        ;
                                 7E         07   A6  9A 000EF          MOVZBL  7(KP), -(SP)                              ;
                                             04  11  000F3             BRB     7$                                        ;
                                 7E         06   A6  9A 000F5 6$:      MOVZBL  6(KP), -(SP)                              ; 0892
                                            1C   AE  9F 000F9 7$:      PUSHAB  H                                         ; 0891
                          0000G  CF             04  FB 000FC          CALLS   #4, ANL$2BUCKET_HEADER                    ;
                                 7E             01  CE 00101 8$:      MNEGL   #1, -(SP)                                 ; 0893
                                            14   AE  9F 00104          PUSHAB  H                                         ;
                          0000G  CF             02  FB 00107          CALLS   #2, ANL$BUCKET                            ;
50            0D   A9                01       01  EF 0010C          EXTZV   #1, #1, 13(HP), R0                        ; 0899
                                 50             6E  C0 00112          ADDL2   (SP), R0                                  ;
                                 1B             50  E9 00115          BLBC    R0, 9$                                    ;
                                            2C   AE  DD 00118          PUSHL   F+4                                       ; 0900
                                 00000000G  8F  DD 0011B             PUSHL   #ANLRMS$_BADBKTROOTBIT                    ;
                          0000G  CF             02  FB 00121          CALLS   #2, ANL$FORMAT_ERROR                      ;
                                 00000000G  8F  DD 00126             PUSHL   #ANLRMS$_UNWIND                           ; 0901
                    00000000G  00               01  FB 0012C          CALLS   #1, LIB$SIGNAL                            ;
                             30   AE             0E  D0 00133 9$:      MOVL    #14, B+8                                  ; 0908
                                            3C   AE  D4 00137          CLRL    B+20                                      ; 0909
                   58        34   AE         30  AE  C1 0013A 10$:     ADDL3   B+8, B+12, RP                             ; 0917
                                 25          04  AE  E9 00140          BLBC    4(SP), 11$                                ; 0918
                   51        3C   AE             01  C1 00144          ADDL3   #1, B+20, R1                              ; 0919
50            0D   A9                02       03  EF 00149          EXTZV   #3, #2, 13(HP), R0                        ;
                                 50             02  C0 0014F          ADDL2   #2, R0                                    ;
                                 50             51  C4 00152          MULL2   R1, R0                                    ;
                   50        38   AE             50  C3 00155          SUBL3   R0, B+16, R0                              ;
                                 57         FC  A0  9E 0015A          MOVAB   -4(R0), VP                                ;
                                 07         04  AE  E9 0015E          BLBC    4(SP), 11$                                ; 0925
                                 50       0000G CF  9E 00162          MOVAB   ANL$3INDEX_RECORD, R0                     ;
                                             05  11  00167             BRB     12$                                       ;
                                 50       0000G CF  9E 00169 11$:     MOVAB   ANL$2INDEX_RECORD, R0                     ;
```

```
                                                7E  D4 0016E 12$:    CLRL    -(SP)                                   ; 0926
                                                5B  DD 00170        PUSHL   R11
                                             30 AE  9F 00172        PUSHAB  B
                                          60    03  FB 00175        CALLS   #3, (R0)
                                       0C AE    50  90 00178        MOVB    R0, ANOTHER_RECORD
                                          21    04 AE  E9 0017C     BLBC    4(SP), 17$                              ; 0932
                50      0D A9               02  03  EF 00180        EXTZV   #3, #2, 13(HP), R0                      ; 0933
                                 02           00  50  CF 00186     CASEL   R0, #0, #2
                              0012         000B      0006  0018A 13$:  .WORD  14$-13$,-
                                                                            15$-13$,-
                                                                            16$-13$
                                          5A        67  3C 00190 14$:  MOVZWL  (VP), DOWN_VBN                      ; 0934
                                                    32  11 00193     BRB     23$
                5A          67              18      00  EF 00195 15$:  EXTZV   #0, #24, (VP), DOWN_VBN             ; 0935
                                                    2B  11 0019A     BRB     23$
                                          5A        67  D0 0019C 16$:  MOVL    (VP), DOWN_VBN                      ; 0936
                                                    26  11 0019F     BRB     23$                                 ; 0933
                50          68              02      00  EF 001A1 17$:  EXTZV   #0, #2, (RP), R0                   ; 0940
                                 02           00  50  CF 001A6     CASEL   R0, #0, #2
                              0014         000C      0006  001AA 18$:  .WORD  19$-18$,-
                                                                            20$-18$,-
                                                                            21$-18$
                                          5A    01  A8  3C 001B0 19$:  MOVZWL  1(RP), DOWN_VBN                     ; 0941
                                                    0C  11 001B4     BRB     22$
                5A      01  A8              18      00  EF 001B6 20$:  EXTZV   #0, #24, 1(RP), DOWN_VBN            ; 0942
                                                    04  11 001BC     BRB     22$
                                          5A    01  A8  D0 001BE 21$:  MOVL    1(RP), DOWN_VBN                     ; 0943
                                          58    03 A048 9E 001C2 22$:  MOVAB   3(R0)[RP], RP                       ; 0945
                                          02    0C  AC  D1 001C7 23$:  CMPL    LEVEL, #2                           ; 0950
                                                    12  1F 001CB     BLSSU   24$
                     50     0C  AC                  01  C3 001CD     SUBL3   #1, LEVEL, R0                       ; 0955
                                                    50  DD 001D2     PUSHL   R0
                                             7E             5A  7D 001D4     MOVQ    DOWN_VBN, -(SP)
                                       FE24  CF             03  FB 001D7     CALLS   #3, ANL$IDX_CHECK_KEY_STUFF
                                                  0115      31  001DC     BRW     39$
                                                08  AE      D4  001DF 24$:  CLRL    8(SP)                           ; 0967
                                          01    0C  AC      D1  001E2     CMPL    LEVEL, #1
                                                    03      12  001E6     BNEQ    25$
                                                08  AE      D6  001E8     INCL    8(SP)
                50      0D  A9                  01          00  EF 001EB 25$:  EXTZV   #0, #1, 13(HP), R0          ; 0968
                                                50          50  D2 001F1     MCOML   R0, R0
                                          08    AE          50  CA 001F4     BICL2   R0, 8(SP)
                                          51    0C  AE      9A  001F8     MOVZBL  ANOTHER_RECORD, R1                  ; 0969
                                          08    AE          51  CA 001FC     BICL2   R1, 8(SP)
                18          00                  6E          00  2C 00200     MOVC5   #0, (SP), #0, #24, R            ; 0978
                                                10  AE          00205
                                          10    AE  0000'   CF  7D 00207 26$:  MOVQ    F, T                        ; 0990
                                          18    AE  0000'   CF  D0 0020D     MOVL    F+8, T+8
                                          24    AE  0000'   CF  D0 00213     MOVL    F+20, T+20
                                                            7E  D4 00219     CLRL    -(SP)
                                                      14    AE  9F 0021B     PUSHAB  T
                                             0000G    CF      02  FB 0021E     CALLS   #2, ANL$BUCKET
                                                17          04  AE  E9 00223     BLBC    4(SP), 27$                  ; 0991
                                                            7E  7C 00227     CLRQ    -(SP)                       ; 0992
                7E      10  A6                  01          00  EF 00229     EXTZV   #0, #1, 16(KP), -(SP)       ; 0993
                                                7E      15  A6  9A 0022F     MOVZBL  21(KP), -(SP)               ; 0992
                                                      0000' CF  9F 00233     PUSHAB  D
```

```
                0000G  CF              05  FB 00237          CALLS   #5, ANL$3BUCKET_HEADER                ;
                                       0F  11 0023C          BRB     28$                                  ;
                                    7E 7C  0023E 27$:        CLRQ    -(SP,                                ; 0995
                       7E       08  A6  9A  00240          MOVZBL  8(KP), -(SP)                         ;
                             0000'  CF  9F  00244          PUSHAB  D                                    ;
                0000G  CF           04  FB  00248          CALLS   #4, ANL$2BUCKET_HEADER               ;
                       52           50  90  0024D 28$:     MOVB    R0, ANOTHER_BUCKET                   ; 0991
                       50       1C  AE  D0  00250          MOVL    R+12, HP                             ; 1002
                       0E       04  A0  B1  00254          CMPW    4(HP), #14                           ; 1003
                                    39  1B  00258          BLEQU   34$                                  ;
                    18 AE           0E  D0  0025A          MOVL    #14, R+8                             ; 1004
                       13       04  AE  E9  0025E 29$:     BLBC    4(SP), 31$                           ; 1007
                                15  A6  95  00262          TSTB    21(KP)                               ;
                                    07  12  00265          BNEQ    30$                                  ;
                       50   0000G  CF  9E  00267          MOVAB   ANL$3PRIMARY_DATA_RECORD, R0         ;
                                    18  11  0026C          BRB     33$                                  ;
                       50   0000G  CF  9E  0026E 30$:     MOVAB   ANL$3SIDR_RECORD, R0                 ;
                                    11  11  00273          BRB     33$                                  ;
                                15  A6  95  00275 31$:     TSTB    21(KP)                               ; 1012
                                    07  12  00278          BNEQ    32$                                  ;
                       50   0000G  CF  9E  0027A          MOVAB   ANL$2PRIMARY_DATA_RECORD, R0         ;
                                    05  11  0027F          BRB     33$                                  ;
                       50   0000G  CF  9E  00281 32$:     MOVAB   ANL$2SIDR_RECORD, R0                 ;
                                    7E  D4  00286 33$:     CLRL    -(SP)                                ; 1015
                                    5B  DD  00288          PUSHL   R11                                  ;
                                18  AE  9F  0028A          PUSHAB  R                                    ;
                       60           03  FB  0028D          CALLS   #3, (R0)                             ;
                       CB           50  E8  00290          BLBS    R0, 29$                              ;
                       1F   0000'  CF  F4  00293 34$:     SOBGEQ  BUCKET_COUNT, 35$                    ; 1022
                       7E       15  A6  9A  00298          MOVZBL  21(KP), -(SP)                        ; 1023
                                18  AE  DD  0029C          PUSHL   R+4                                  ;
                         00000000G  8F  DD  0029F          PUSHL   #ANLRMS$_BKTLOOP                     ;
                0000G  CF           03  FB  002A5          CALLS   #3, ANL$FORMAT_ERROR                 ;
                         00000000G  8F  DD  002AA          PUSHL   #ANLRMS$_UNWIND                      ; 1024
             00000000G  00          01  FB  002B0          CALLS   #1, LIB$SIGNAL                       ;
                       05       08  AE  E9  002B7 35$:     BLBC    8(SP), 36$                           ; 1033
                       28           52  E8  002BB          BLBS    ANOTHER_BUCKET, 37$                  ; 1034
                                    29  11  002BE          BRB     38$                                  ;
                       5A       14  AE  D1  002C0 36$:     CMPL    R+4, DOWN_VBN                        ; 1036
                                    23  13  002C4          BEQL    38$                                  ;
                       1D           52  E8  002C6          BLBS    ANOTHER_BUCKET, 37$                  ; 1039
                                    5A  DD  002C9          PUSHL   DOWN_VBN                             ; 1040
                                30  AE  DD  002CB          PUSHL   B+4                                  ;
                         00000000G  8F  DD  002CE          PUSHL   #ANLRMS$_MISSINGBKT                  ;
                0000G  CF           03  FB  002D4          CALLS   #3, ANL$FORMAT_ERROR                 ;
                         00000000G  8F  DD  002D9          PUSHL   #ANLRMS$_UNWIND                      ; 1041
             00000000G  00          01  FB  002DF          CALLS   #1, LIB$SIGNAL                       ;
                           FF1E     31  002E6 37$:         BRW     26$                                  ; 0978
                       7E           01  CE  002E9 38$:     MNEGL   #1, -(SP)                            ; 1046
                                14  AE  9F  002EC          PUSHAB  R                                    ;
                0000G  CF           02  FB  002EF          CALLS   #2, ANL$BUCKET                       ;
                       03       0C  AE  E9  002F4 39$:     BLBC    ANOTHER_RECORD, 40$                  ; 1051
                             FE3F  31  002F8              BRW     10$                                  ;
                       7E           01  CE  002FB 40$:     MNEGL   #1, -(SP)                            ; 1054
                                2C  AE  9F  002FE          PUSHAB  B                                    ;
                0000G  CF           02  FB  00301          CALLS   #2, ANL$BUCKET                       ;
                       0C           6E  E9  00306          BLBC    (SP), 41$                            ; 1055
```

```
                                 7E           01  CE 00309          MNEGL   #1, -(SP)                                ; 1056
                                        0000'  CF  9F 0030C          PUSHAB  D                                        ;
                              0000G  CF         02  FB 00310          CALLS   #2, ANL$BUCKET                           ;
                                              04 00315 41$:          RET                                              ; 1060

; Routine Size:  790 bytes,    Routine Base:  $CODE$ + 0172

;  561          1061  1
;  562          1062  0 end eludom


                                                                     .EXTRN  LIB$SIGNAL

;                                    PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                         1160  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                            28  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                                  Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1          18619        33         0        1000      00:01.7



;                                         COMMAND QUALIFIERS

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

; Size:          1160 code + 28 data bytes
; Run Time:         00:23.7
; Elapsed Time:     01:23.2
; Lines/CPU Min:     2687
; Lexemes/CPU-Min:  18177
; Memory Used:   302 pages
; Compilation Complete
```

RMSINTER
LIS

RMSCHECKA
LIS

RMSFDL
LIS

RMSCHECKB
LIS

RMSINPUT
LIS

RMSMSG
LIS