```
                                        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
```

FILEID**RMS2IDX

```
RRRRRRRR    MM      MM    SSSSSSSS    222222      IIIIII    DDDDDDDD    XX      XX
RRRRRRRR    MM      MM    SSSSSSSS    222222      IIIIII    DDDDDDDD    XX      XX
RR      RR  MMMM  MMMM  SS          22      22      II      DD      DD  XX      XX
RR      RR  MMMM  MMMM  SS          22      22      II      DD      DD  XX      XX
RR      RR  MM  MM  MM  SS                  22      II      DD      DD    XX  XX
RR      RR  MM  MM  MM  SS                  22      II      DD      DD    XX  XX
RRRRRRRR    MM      MM    SSSSSS          22        II      DD      DD      XX
RRRRRRRR    MM      MM    SSSSSS          22        II      DD      DD      XX
RR  RR      MM      MM          SS      22          II      DD      DD    XX  XX
RR  RR      MM      MM          SS      22          II      DD      DD    XX  XX
RR    RR    MM      MM          SS    22            II      DD      DD  XX      XX      ....
RR    RR    MM      MM          SS    22            II      DD      DD  XX      XX      ....
RR      RR  MM      MM  SSSSSSSS    2222222222    IIIIII    DDDDDDDD    XX      XX      ....
RR      RR  MM      MM  SSSSSSSS    2222222222    IIIIII    DDDDDDDD    XX      XX      ....


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
```

```
0 %title 'RMS2IDX - Analyze Things for Prolog 2 Indexed Files'
0         module rms2idx  (
1                         ident='V04-000') = begin
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! Facility:      VAX/VMS Analyze Facility, Analyze Things for Prolog 2
1 !
1 ! Abstract:      This module is responsible for analyzing various structures
1 !                in prolog 2 indexed files.  It also includes those routines
1 !                that are common to prolog 2 and 3.
1 !
1 !
1 ! Environment:
1 !
1 ! Author: Paul C. Anagnostopoulos, Creation Date: 11 March 1981
1 !
1 ! Modified By:
1 !
1 !         V03-005 PCA1012         Paul C. Anagnostopoulos  6-Apr-1983
1 !                 Change the bucket size check so that it uses the new
1 !                 literal value BKT$C_MAXBKTSIZ.  The maximum bucket size
1 !                 was increased, so a literal value was a good idea.
1 !                 Add code to handle the new total area allocation field
1 !                 in the area descriptor.
1 !
1 !         V03-004 PCA1011         Paul C. Anagnostopoulos  1-Apr-1983
1 !                 Change the message prefix to ANLRMS$_ to ensure that
1 !                 message symbols are unique across all ANALYZEs.  This
1 !                 is necessitated by the new merged message files.
1 !
1 !         V03-003 PCA1001         Paul C. Anagnostopoulos 12-Oct-1982
1 !                 Clean up this module to make it more consistent with
```

```
;   58          0058  1 !                 the prologue 3 stuff in RMS3IDX, particularly where
;   59          0059  1 !                 SIDRs are concerned.  Remove all of the alignment
;   60          0060  1 !                 information from the area descriptor display.  Add the
;   61          0061  1 !                 new quadword key data types.
;   62          0062  1 !
;   63          0063  1 !         V03-002 PCA0001         Paul Anagnostopoulos    16-Mar-1982
;   64          0064  1 !                 Remove logic for prologue 3 data type array in key
;   65          0065  1 !                 descriptor.  It's been decommitted for V3A.
;   66          0066  1 !
;   67          0067  1 !         V03-001 PCA0002         Paul Anagnostopoulos    16-Mar-1982
;   68          0068  1 !                 Don't display root and data bucket VBNs if the index
;   69          0069  1 !                 is not initialized.
;   70          0070  1 !--
```

```
:    72          0071  1 %sbttl 'Module Declarations'
:    73          0072  1 !
:    74          0073  1 ! Libraries and Requires:
:    75          0074  1 !
:    76          0075  1
:    77          0076  1 library 'lib';
:    78          0077  1 require 'rmsreq';
:    79          0586  1
:    80          0587  1 !
:    81          0588  1 ! Table of Contents:
:    82          0589  1 !
:    83          0590  1
:    84          0591  1 forward routine
:    85          0592  1         anl$idx_prolog: novalue,
:    86          0593  1         anl$area_descriptor: novalue,
:    87          0594  1         anl$key_descriptor,
:    88          0595  1         anl$2bucket_header,
:    89          0596  1         anl$2index_record,
:    90          0597  1         anl$2primary_data_record,
:    91          0598  1         anl$2format_primary_key: novalue,
:    92          0599  1         anl$2sidr_record,
:    93          0600  1         anl$2sidr_pointer;
:    94          0601  1
:    95          0602  1 !
:    96          0603  1 ! External References:
:    97          0604  1 !
:    98          0605  1
:    99          0606  1 external routine
:   100          0607  1         anl$bucket,
:   101          0608  1         anl$bucket_callback,
:   102          0609  1         anl$check_flags,
:   103          0610  1         anl$data_callback,
:   104          0611  1         anl$format_error,
:   105          0612  1         anl$format_flags,
:   106          0613  1         anl$format_hex,
:   107          0614  1         anl$format_line,
:   108          0615  1         anl$format_skip,
:   109          0616  1         anl$index_callback,
:   110          0617  1         anl$prepare_quoted_string;
:   111          0618  1
:   112          0619  1 external
:   113          0620  1         anl$gb_mode: byte,
:   114          0621  1         anl$gl_fat: ref block[,byte],
:   115          0622  1         anl$gw_prolog: word;
:   116          0623  1
:   117          0624  1 !
:   118          0625  1 ! Own Variables:
:   119          0626  1 !
```

```
;  121        0627  1 %sbttl 'ANL$IDX_PROLOG - Format and Check an Indexed File Prolog'
;  122        0628  1 !++
;  123        0629  1 ! Functional Description:
;  124        0630  1 !       This routine is reponsible for formatting a report and checking
;  125        0631  1 !       the prolog of an indexed file.
;  126        0632  1 !
;  127        0633  1 ! Formal Parameters:
;  128        0634  1 !       prolog_bsd      A BSD describing the prolog.
;  129        0635  1 !       report          A boolean, true if we are to print a report.
;  130        0636  1 !       indent_level    The indentation level of the report.
;  131        0637  1 !
;  132        0638  1 ! Implicit Inputs:
;  133        0639  1 !       global data
;  134        0640  1 !
;  135        0641  1 ! Implicit Outputs:
;  136        0642  1 !       global data
;  137        0643  1 !
;  138        0644  1 ! Returned Value:
;  139        0645  1 !       none
;  140        0646  1 !
;  141        0647  1 ! Side Effects:
;  142        0648  1 !
;  143        0649  1 !--
;  144        0650  1
;  145        0651  1
;  146        0652  2 global routine anl$idx_prolog(prolog_bsd,report,indent_level): novalue = begin
;  147        0653  2
;  148        0654  2 bind
;  149        0655  2         p = .prolog_bsd: bsd;
;  150        0656  2
;  151        0657  2 local
;  152        0658  2         sp: ref block[,byte];
;  153        0659  2
;  154        0660  2
;  155        0661  2 ! We can start right off and format the prolog if requested.  Begin with
;  156        0662  2 ! a nice heading
;  157        0663  2
;  158        0664  2 sp = .p[bsd$l_bufptr];
;  159        0665  3 if .report then (
;  160        0666  3         anl$format_line(3,.indent_level,anlrms$_idxprolog);
;  161        0667  3         anl$format_skip(0);
;  162        0668  3
;  163        0669  3         ! Format the first area VBN and number of areas.
;  164        0670  3
;  165        0671  3         anl$format_line(0,.indent_level+1,anlrms$_idxproareas,.sp[plg$b_amax],.sp[plg$b_avbn]);
;  166        0672  3
;  167        0673  3         ! Format the prolog version number.
;  168        0674  3
;  169        0675  3         anl$format_line(0,.indent_level+1,anlrms$_prologver,.sp[plg$w_ver_no]);
;  170        0676  2 );
;  172        0677  2 ! Now we can check the prolog.  Make sure the area information is reasonable.
;  173        0678  2
;  174        0679  2 if .sp[plg$b_avbn] lssu 2 or
;  175        0680  2    .sp[plg$b_amax] eqlu 0        then
;  176        0681  2         anl$format_error(anlrms$_badarearoot,.p[bsd$l_vbn]);
;  177        0682  2
;  178        0683  2 return;
;  179        0684  2
;  180        0685  1 end;


                                                              .TITLE  RMS2IDX RMS2IDX - Analyze Things for Prolog 2 I
;                                                                             ndexed F
                                                              .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  ANI RMS$_FDLIDXLENMEAN
.EXTRN  ANLRMS$_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$BUCKET, ANL$BUCKET_CALLBACK
                                                                .EXTRN  ANL$CHECK_FLAGS
                                                                .EXTRN  ANL$DATA_CALLBACK
                                                                .EXTRN  ANL$FORMAT_ERROR
                                                                .EXTRN  ANL$FORMAT_FLAGS
                                                                .EXTRN  ANL$FORMAT_HEX, ANL$FORMAT_LINE
                                                                .EXTRN  ANL$FORMAT_SKIP
                                                                .EXTRN  ANL$INDEX_CALLBACK
                                                                .EXTRN  ANL$PREPARE_QUOTED_STRING
                                                                .EXTRN  ANL$GB_MODE, ANL$GL_FAT
                                                                .EXTRN  ANL$GW_PROLOG

                                                                .PSECT  $CODE$,NOWRT,2

                                                    003C 00000  .ENTRY  ANL$IDX_PROLOG, Save R2,R3,R4,R5                   ; 0652
                        55       0000G  CF       9E 00002       MOVAB   ANL$FORMAT_LINE, R5                                 ;
                        54          04  AC       D0 00007       MOVL    PROLOG_BSD, R4                                      ; 0655
                        52          0C  A4       D0 0000B       MOVL    12(R4), SP                                          ; 0664
                        40          08  AC       E9 0000F       BLBC    REPORT, 1$                                          ; 0665
                             00000000G  8F       DD 00013       PUSHL   #ANLRMS$_IDXPROLOG                                  ; 0666
                                    0C  AC       DD 00019       PUSHL   INDENT_LEVEL                                        ;
                                        03       DD 0001C       PUSHL   #3                                                  ;
                        65              03       FB 0001E       CALLS   #3, ANL$FORMAT_LINE                                 ;
                                        7E       D4 00021       CLRL    -(SP)                                               ; 0667
              0000G     CF              01       FB 00023       CALLS   #1, ANL$FORMAT_SKIP                                 ;
                        7E          66  A2       9A 00028       MOVZBL  102(SP), -(SP)                                      ; 0671
                        7E          67  A2       9A 0002C       MOVZBL  103(SP), -(SP)                                      ;
                             00000000G  8F       DD 00030       PUSHL   #ANLRMS$_IDXPROAREAS                                ;
        53         0C   AC              01       C1 00036       ADDL3   #1, INDENT_LEVEL, R3                                ;
                                        53       DD 0003B       PUSHL   R3                                                  ;
                                        7E       D4 0003D       CLRL    -(SP)                                               ;
                        65              05       FB 0003F       CALLS   #5, ANL$FORMAT_LINE                                 ;
                        7E          74  A2       3C 00042       MOVZWL  116(SP), -(SP)                                      ; 0675
                             00000000G  8F       DD 00046       PUSHL   #ANLRMS$_PROLOGVER                                  ;
                                        53       DD 0004C       PUSHL   R3                                                  ;
                                        7E       D4 0004E       CLRL    -(SP)                                               ;
                        65              04       FB 00050       CALLS   #4, ANL$FORMAT_LINE                                 ;
```

```
                     02         66  A2  91 00053 1$:     CMPB    102(SP), #2                          ; 0679
                                    05  1F 00057         BLSSU   2$                                   ;
                                67  A2  95 00059         TSTB    103(SP)                              ; 0680
                                    0E  12 0005C         BNEQ    3$                                   ;
                                04  A4  DD 0005E 2$:     PUSHL   4(R4)                                ; 0681
                         00000000G  8F  DD 00061         PUSHL   #ANLRMS$_BADAREAROOT                 ;
                 0000G  CF          02  FB 00067         CALLS   #2, ANL$FORMAT_ERROR                 ;
                                        04 0006C 3$:     RET                                          ; 0685

; Routine Size:  109 bytes,     Routine Base:  $CODE$ + 0000
```

```
1 %sbttl 'ANL$AREA_DESCRIPTOR: Check and Format an Area Descriptor'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for checking the content of an area
1 !       descriptor and optionally printing a formatted report of it.
1 !
1 ! Formal Parameters:
1 !       the_bsd         The address of a BSD describing the area descriptor.
1 !                       We update the BSD to describe the next one.
1 !       area_id         Alleged ID of this area.
1 !       report          A boolean, true if we are to print a report.
1 !       indent_level    The indentation level of the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$area_descriptor(the_bsd,area_id,report,indent_level): novalue = begin
2
2 bind
2       b = .the_bsd: bsd;
2
2 local
2       sp: ref block[,byte],
2       next_id: long;
2
2 ! Since we know we have 64 bytes in the block, we don't have to check that
2 ! things actually fit in the block.
2 ! So we can start right off and format the report if requested. Begin with
2 ! a nice header containing the area id.
2
2 sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
3 if .report then (
3       anl$format_line(4,.indent_level,anlrms$_idxarea,.sp[area$b_areaid],
3                       .b[bsd$l_vbn],.b[bsd$l_offset]);
3       anl$format_skip(0);
3
3       ! Format the area bucket size.
3
3       anl$format_line(0,.indent_level+1,anlrms$_idxareabktsz,.sp[area$b_arbktsz]);
3
3       ! Format the reclaimed bucket pointer.  It's only used for prolog 3.
3
3       if .anl$gw_prolog eqlu plg$c_ver_3 then
3               anl$format_line(0,.indent_level+1,anlrms$_idxarearecl,.sp[area$l_avail]);
3
;   239          0743  3          ! Format the info describing how much of the current extent has been
;   240          0744  3          ! used up.
;   241          0745  3
;   242          0746  3          anl$format_line(0,.indent_level+1,anlrms$_idxareaused,.sp[area$l_cvbn],
;   243          0747  3                          .sp[area$l_cnblk],.sp[area$l_used],.sp[area$l_nxtvbn]);
;   244          0748  3
;   245          0749  3          ! Format the info describing the next extent, if present.
;   246          0750  3
;   247          0751  3          if .sp[area$l_nxt] nequ 0 or .sp[area$l_nxblk] nequ 0 then
;   248          0752  3                  anl$format_line(0,.indent_level+1,anlrms$_idxareanext,
;   249          0753  3                                  .sp[area$l_nxt],.sp[area$l_nxblk]);
;   250          0754  3
;   251          0755  3          ! Format the default extend quantity.
;   252          0756  3
;   253          0757  3          anl$format_line(0,.indent_level+1,anlrms$_idxareaqty,.sp[area$w_deq]);
;   254          0758  3
;   255          0759  3          ! If an extent has been allocated but the total allocation is zero,
;   256          0760  3          ! then this file was created before the total allocation field
;   257          0761  3          ! existed.  Just put out a comment.  Otherwise, we can put out the
;   258          0762  3          ! total area allocation.
;   259          0763  3
;   260          0764  3          if .sp[area$l_cvbn] nequ 0 and .sp[area$l_total_alloc] eqlu 0 then
;   261          0765  3                  anl$format_line(0,.indent_level+1,anlrms$_idxareanoalloc)
;   262          0766  3          else
;   263          0767  3                  anl$format_line(0,.indent_level+1,anlrms$_idxareaalloc,.sp[area$l_total_alloc]);
;   264          0768  2 );
;  266          0769  2 ! Now we are going to check the contents of the area descriptor.  This is
;  267          0770  2 ! a fairly rigorous test, but doesn't check anything that requires looking
;  268          0771  2 ! at other structures.
;  269          0772  2
;  270          0773  2 ! Start be ensuring that the first two bytes area unused.
;  271          0774  2
;  272          0775  2 if .sp[0,0,16,0] nequ 0 then
;  273          0776  2         anl$format_error(anlrms$_badarea1st2,.b[bsd$l_vbn],.area_id);
;  274          0777  2
;  275          0778  2 ! Make sure the area ID is correct
;  276          0779  2
;  277          0780  2 if .sp[area$b_areaid] nequ .area_id then
;  278          0781  2         anl$format_error(anlrms$_badareaid,.b[bsd$l_vbn],.sp[area$b_areaid],.area_id);
;  279          0782  2
;  280          0783  2 ! Check the area bucket size.
;  281          0784  2
;  282          0785  2 if .sp[area$b_arbktsz] lssu 1 or .sp[area$b_arbktsz] gtru bkt$c_maxbktsiz then
;  283          0786  2         anl$format_error(anlrms$_badareabktsize,.b[bsd$l_vbn],.sp[area$b_arbktsz],.area_id);
;  284          0787  2
;  285          0788  2 ! We ought to check the current extent information at this point, but no
;  286          0789  2 ! one can tell me how it is used.  So the code is commented out for now,
;  287          0790  2 ! and a !!!TEMP!!! flag marks the situation.
;  288          0791  2
;  289          0792  2 !if .sp[area$l_used] gtru .sp[area$l_cnblk] or
;  290          0793  2 !   .sp[area$l_cvbn]+.sp[area$l_used] nequ .sp[area$l_nxtvbn] then
;  291          0794  2 !       anl$format_error(anlrms$_badareaused,.b[bsd$l_vbn]);
;  292          0795  2
;  293          0796  2 ! The two items describing the next extent must both be absent or both present.
;  294          0797  2
;  295          0798  2 if .sp[area$l_nxt] eqlu 0 xor .sp[area$l_nxblk] eqlu 0 then
;  296          0799  2         anl$format_error(anlrms$_badareanext,.b[bsd$l_vbn],.area_id);
;   298          0800  2 ! Now we want to advance on to the next area descriptor, if there is one.
;   299          0801  2 ! Begin by reading in the first prolog block.
;   300          0802  2
;   301          0803  2 b[bsd$l_vbn] = 1;
;   302          0804  2 anl$bucket(b,0);
;   303          0805  2
;   304          0806  2 ! Determine the id of the next area, or this area again if it's the last one.
;   305          0807  2
;   306          0808  2 sp = .b[bsd$l_bufptr];
;   307          0809  2 next_id = minu(.area_id+1,.sp[plg$b_amax]-1);
;   308          0810  2
;   309          0811  2 ! Now read in the appropriate block and set the offset.
;   310          0812  2
;   311          0813  2 b[bsd$l_vbn] = .sp[plg$b_avbn] + .next_id / (512/area$c_bln);
;   312          0814  2 b[bsd$l_offset] = .next_id mod (512/area$c_bln) * area$c_bln;
;   313          0815  2 anl$bucket(b,0);
;   314          0816  2
;   315          0817  2 return;
;   316          0818  2
;   317          0819  1 end;
```

```
                                             007C 00000            .ENTRY   ANL$AREA_DESCRIPTOR, Save R2,R3,R4,R5,R6       ; 0713
                           56       0000G  CF   9E 00002            MOVAB    ANL$FORMAT_ERROR, R6                           ;
                           55       0000G  CF   9E 00007            MOVAB    ANL$FORMAT_LINE, R5                            ;
                           53          04  AC   D0 0000C            MOVL     THE_BSD, R3                                    ; 0716
             52        0C  A3          08  A3   C1 00010            ADDL3    8(R3), 12(R3), SP                              ; 0728
                           03          0C  AC   E8 00016            BLBS     REPORT, 1$                                     ; 0729
                                         00B4   31 0001A            BRW      6$                                             ;
                           7E          04  A3   7D 0001D 1$:        MOVQ     4(R3), -(SP)                                   ; 0731
                           7E          02  A2   9A 00021            MOVZBL   2(SP), -(SP)                                   ; 0730
                              00000000G  8F   DD 00025            PUSHL    #ANLRMS$_IDXAREA                               ;
                                       10  AC   DD 0002B            PUSHL    INDENT_LEVEL                                   ;
                                           04   DD 0002E            PUSHL    #4                                             ;
                           65              06   FB 00030            CALLS    #6, ANL$FORMAT_LINE                            ;
                                           7E   D4 00033            CLRL     -(SP)                                          ; 0732
                     0000G CF              01   FB 00035            CALLS    #1, ANL$FORMAT_SKIP                            ;
                           7E          03  A2   9A 0003A            MOVZBL   3(SP), -(SP)                                   ; 0736
                              00000000G  8F   DD 0003E            PUSHL    #ANLRMS$_IDXAREABKTSZ                          ;
             54        10  AC              01   C1 00044            ADDL3    #1, INDENT_LEVEL, R4                           ;
                                           54   DD 00049            PUSHL    R4                                             ;
                                           7E   D4 0004B            CLRL     -(SP)                                          ;
                           65              04   FB 0004D            CALLS    #4, ANL$FORMAT_LINE                            ;
                           03       0000G  CF   B1 00050            CMPW     ANL$GW_PROLOG, #3                              ; 0740
                                           10   12 00055            BNEQ     2$                                             ;
                                       08  A2   DD 00057            PUSHL    8(SP)                                          ; 0741
                              00000000G  8F   DD 0005A            PUSHL    #ANLRMS$_IDXAREARECL                           ;
                                           54   DD 00060            PUSHL    R4                                             ;
                                           7E   D4 00062            CLRL     -(SP)                                          ;
                           65              04   FB 00064            CALLS    #4, ANL$FORMAT_LINE                            ;
                           7E          14  A2   7D 00067 2$:        MOVQ     20(SP), -(SP)                                  ; 0747
                           7E          0C  A2   7D 0006B            MOVQ     12(SP), -(SP)                                  ; 0746
                              00000000G  8F   DD 0006F            PUSHL    #ANLRMS$_IDXAREAUSED                           ;
                                           54   DD 00075            PUSHL    R4                                             ;
```

```
                                                 7E  D4 00077          CLRL    -(SP)                                  ;
                          65                     07  FB 00079          CALLS   #7, ANL$FORMAT_LINE                    ;
                                         1C      A2  D5 0007C          TSTL    28(SP)                                 ; 0751
                                                 05  12 0007F          BNEQ    3$                                     ;
                                         20      A2  D5 0008'          TSTL    32(SP)                                 ;
                                                 11  13 00084          BEQL    4$                                     ;
                          7E             1C      A2  7D 00086 3$:      MOVQ    28(SP), -(SP)                          ; 0753
                                  00000000G      8F  DD 0008A          PUSHL   #ANLRMS$_IDXAREANEXT                   ; 0752
                                                 54  DD 00090          PUSHL   R4                                     ;
                                                 7E  D4 00092          CLRL    -(SP)                                  ;
                          65                     05  FB 00094          CALLS   #5, ANL$FORMAT_LINE                    ;
                          7E             24      A2  3C 00097 4$:      MOVZWL  36(SP), -(SP)                          ; 0757
                                  00000000G      8F  DD 0009B          PUSHL   #ANLRMS$_IDXAREAQTY                    ;
                                                 54  DD 000A1          PUSHL   R4                                     ;
                                                 7E  D4 000A3          CLRL    -(SP)                                  ;
                          65                     04  FB 000A5          CALLS   #4, ANL$FORMAT_LINE                    ;
                                         0C      A2  D5 000A8          TSTL    12(SP)                                 ; 0764
                                                 14  13 000AB          BEQL    5$                                     ;
                                         32      A2  D5 000AD          TSTL    50(SP)                                 ;
                                                 0F  12 000B0          BNEQ    5$                                     ;
                                  00000000G      8F  DD 000B2          PLSHL   #ANLRMS$_IDXAREANOALLOC                ; 0765
                                                 54  DD 000B8          PUJHL   R4                                     ;
                                                 7E  D4 000BA          CLRL    -(SP)                                  ;
                          65                     03  FB 000BC          CALLS   #3, ANL$FORMAT_LINE                    ;
                                                 10  11 000BF          BRB     6$                                     ;
                                         32      A2  DD 000C1 5$:      PUSHL   50(SP)                                 ; 0767
                                  00000000G      8F  DD 000C4          PUSHL   #ANLRMS$_IDXAREAALLOC                  ;
                                                 54  DD 000CA          PUSHL   R4                                     ;
                                                 7E  D4 000CC          CLRL    -(SP)                                  ;
                          65                     04  FB 000CE          CALLS   #4, ANL$FORMAT_LINE                    ;
                                                 62  B5 000D1 6$:      TSTW    (SP)                                   ; 0775
                                                 0F  13 000D3          BEQL    7$                                     ;
                                         08      AC  DD 000D5          PUSHL   AREA_ID                                ; 0776
                                         04      A3  DD 000D8          PUSHL   4(R3)                                  ;
                                  00000000G      8F  DD 000DB          PUSHL   #ANLRMS$_BADAREA1ST2                   ;
                          66                     03  FB 000E1          CALLS   #3, ANL$FORMAT_ERROR                   ;
                          54             08      AC  D0 000E4 7$:      MOVL    AREA_ID, R4                            ; 0780
    54       02    A2     08                     00  ED 000E8          CMPZV   #0, #8, 2(SP), R4                      ;
                                                 12  13 000EE          BEQL    8$                                     ;
                                                 54  DD 000F0          PUSHL   R4                                     ; 0781
                          7E             02      A2  9A 000F2          MOVZBL  2(SP), -(SP)                           ;
                                         04      A3  DD 000F6          PUSHL   4(R3)                                  ;
                                  00000000G      8F  DD 000F9          PUSHL   #ANLRMS$_BADAREAID                     ;
                          66                     04  FB 000FF          CALLS   #4, ANL$FORMAT_ERROR                   ;
                                         03      A2  95 00102 8$:      TSTB    3(SP)                                  ; 0785
                                                 06  13 00105          BEQL    9$                                     ;
                          3F             03      A2  91 00107          CMPB    3(SP), #63                             ;
                                                 12  1B 0010B          BLEQU   10$                                    ;
                                                 54  DD 0010D 9$:      PUSHL   R4                                     ; 0786
                          7E             03      A2  9A 0010F          MOVZBL  3(SP), -(SP)                           ;
                                         04      A3  DD 00113          PUSHL   4(R3)                                  ;
                                  00000000G      8F  DD 00116          PUSHL   #ANLRMS$_BADAREABKTSIZE                ;
                          66                     04  FB 0011C          CALLS   #4, ANL$FORMAT_ERROR                   ;
                                                 51  D4 0011F 10$:     CLRL    R1                                     ; 0798
                                         1C      A2  D5 00121          TSTL    28(SP)                                 ;
                                                 02  12 00124          BNEQ    11$                                    ;
                                                 51  D6 00126          INCL    R1                                     ;
```

```
                                            50 D4 00128 11$:   CLRL   R0                            ;
                                        20  A2 D5 0012A        TSTL   32(SP)                        ;
                                            02 12 0012D        BNEQ   12$                           ;
                                            50 D6 0012F        INCL   R0                            ;
                                50          51 C0 00131 12$:   ADDL2  R1, R0                        ;
                                0E          50 E9 00134        BLBC   R0, 13$                       ;
                                            54 DD 00137        PUSHL  R4                            ; 0799
                                        04  A3 DD 00139        PUSHL  4(R3)                         ;
                                 00000000G  8F DD 0013C        PUSHL  #ANLRMS$_BADAREANEXT          ;
                                66          03 FB 00142        CALLS  #3, ANL$FORMAT_ERROR          ;
                           04   A3          01 D0 00145 13$:   MOVL   #1, 4(R3)                     ; 0803
                                            7E D4 00149        CLRL   -(SP)                         ; 0804
                                            53 DD 0014B        PUSHL  R3                            ;
                        0000G   CF          02 FB 0014D        CALLS  #2, ANL$BUCKET                ;
                                52      0C  A3 D0 00152        MOVL   12(R3), SP                    ; 0808
                                51      01  A4 9E 00156        MOVAB  1(R4), R1                     ; 0809
                                50      67  A2 9A 0015A        MOVZBL 103(SP), R0                   ;
                                            50 D7 0015E        DECL   R0                            ;
                                50          51 D1 00160        CMPL   R1, R0                        ;
                                            03 1B 00163        BLEQU  14$                           ;
                                51          50 D0 00165        MOVL   R0, R1                        ;
                                50          51 D0 00168 14$:   MOVL   R1, NEXT_ID                   ;
                    51          50          08 C7 0016B        DIVL3  #8, NEXT_ID, R1               ; 0813
                                54      66  A2 9A 0016F        MOVZBL 102(SP), R4                   ;
                04  A3          51          54 C1 00173        ADDL3  R4, R1, 4(R3)                 ;
         7E         00          50          01 7A 00178        EMUL   #1, NEXT_ID, #0, -(SP)        ; 0814
         50         50          8E          08 7B 0017D        EDIV   #8, (SP)+, R0, R0             ;
                08  A3          50          06 78 00182        ASHL   #6, R0, 8(R3)                 ;
                                            7E D4 00187        CLRL   -(SP)                         ; 0815
                                            53 DD 00189        PUSHL  R3                            ;
                        0000G   CF          02 FB 0018B        CALLS  #2, ANL$BUCKET                ;
                                               04 00190        RET                                  ; 0819

; Routine Size:  401 bytes,    Routine Base:  $CODE$ + 006D
;  319     0820  1 %sbttl 'ANL$KEY_DESCRIPTOR - Print and Check a Key Descriptor'
;  320     0821  1 !++
;  321     0822  1 ! Functional Description:
;  322     0823  1 !        This routine is responsible for printing and checking the contents
;  323     0824  1 !        of an indexed file key descriptor.
;  324     0825  1 !
;  325     0826  1 ! Formal Parameters:
;  326     0827  1 !        the_bsd          The address of a BSD describing the key descriptor.
;  327     0828  1 !                         We update it to describe the next one.
;  328     0829  1 !        key_id           The alleged ID of this key.
;  329     0830  1 !        areas            Address of a vector of 256 bytes, one per area.
;  330     0831  1 !                         Contains the bucket size of each area.  Optional.
;  331     0832  1 !        report           A boolean, true if we are to print a report.
;  332     0833  1 !        indent_level     The indentation level of the report.
;  333     0834  1 !
;  334     0835  1 ! Implicit Inputs:
;  335     0636  1 !        global data
;  336     0837  1 !
;  337     0838  1 ! Implicit Outputs:
;  338     0839  1 !        global data
;  339     0840  1 !
;  340     0841  1 ! Returned Value:
;  341     0842  1 !        True if there is another key descriptor, false if not.
;  342     0843  1 !
;  343     0844  1 ! Side Effects:
;  344     0845  1 !
;  345     0846  1 !--
;  346     0847  1
;  347     0848  1
;  348     0849  2 global routine anl$key_descriptor(the_bsd,key_id,areas,report,indent_level) = begin
;  349     0850  2
;  350     0851  2 bind
;  351     0852  2        b = .the_bsd: bsd,
;  352     0853  2        areas_vector = .areas: vector[256,byte];
;  353     0854  2
;  354     0855  2 own
;  355     0856  2        key2_primary_def: vector[6,long] initial(
;  356     0857  2                                        4,
;  357     0858  2                                        uplit byte (%ascic 'KEY$V_DUPKEYS'),
;  358     0859  2                                        0,
;  359     0860  2                                        0,
;  360     0861  2                                        0,
;  361     0862  2                                        uplit byte (%ascic 'KEY$V_INITIDX')
;  362     0863  2                                        ),
;  363     0864  2
;  364     0865  2        key2_secondary_def: vector[6,long] initial(
;  365     0866  2                                        4,
;  366     0867  2                                        uplit byte (%ascic 'KEY$V_DUPKEYS'),
;  367     0868  2                                        uplit byte (%ascic 'KEY$V_CHGKEYS'),
;  368     0869  2                                        uplit byte (%ascic 'KEY$V_NULKEYS'),
;  369     0870  2                                        0,
;  370     0871  2                                        uplit byte (%ascic 'KEY$V_INITIDX')
;  371     0872  2                                        ),
;  372     0873  2
;  373     0874  2        key3_primary_def: vector[9,long] initial(
;  374     0875  2                                        7,
;  375     0876  2                                        uplit byte (%ascic 'KEY$V_DUPKEYS'),
;  2                                     0,
;  2                                     0,
;  2                                     uplit byte (%ascic 'KEY$V_IDX_COMPR'),
;  2                                     uplit byte (%ascic 'KEY$V_INITIDX'),
;  2                                     0,
;  2                                     uplit byte (%ascic 'KEY$V_KEY_COMPR'),
;  2                                     uplit byte (%ascic 'KEY$V_REC_COMPR')
;  2                                     ),
;  2
;  2          key3_secondary_def: vector[8,long] initial(
;  2                                     6,
;  2                                     uplit byte (%ascic 'KEY$V_DUPKEYS'),
;  2                                     uplit byte (%ascic 'KEY$V_CHGKEYS'),
;  2                                     uplit byte (%ascic 'KEY$V_NULKEYS'),
;  2                                     uplit byte (%ascic 'KEY$V_IDX_COMPR'),
;  2                                     uplit byte (%ascic 'KEY$V_INITIDX'),
;  2                                     0,
;  2                                     uplit byte (%ascic 'KEY$V_KEY_COMPR')
;  2                                     );
;  2
;  2  local
;  2          sp: ref block[,byte],
;  2          i: long,
;  2          position: word, size: byte,
;  2          total_size: long, required_record: long;
;  2
;  2  builtin
;  2          nullparameter;
;  2
;  2
;  2  ! This little internal subroutine receives a data type code and returns
;  2  ! the address of an ASCIC string naming the data type.
;  2
;  3  routine data_type_name(code) = begin
;  3
;  3  own
;  3          data_types: vector[8,long] initial(
;  3                                     uplit byte (%ascic 'string'),
;  3                                     uplit byte (%ascic 'signed word'),
;  3                                     uplit byte (%ascic 'unsigned word'),
;  3                                     uplit byte (%ascic 'signed longword'),
;  3                                     uplit byte (%ascic 'unsigned longword'),
;  3                                     uplit byte (%ascic 'packed decimal'),
;  3                                     uplit byte (%ascic 'signed quadword'),
;  3                                     uplit byte (%ascic 'unsigned quadword')
;  3                                     );
;  3
;  4  return  (if .code gtru key$c_max_data then uplit byte (%ascic '???')
;  3                                            else .data_types[.code]);
;  2  end;


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

 53 59 45 4B 50 55 44 5F 56 24 59 45 4B 0D  00000 P.AAA:  .ASCII  <13>\KEY$V_DUPKEYS\           ;
 58 44 49 54 49 4E 49 5F 56 24 59 45 4B 0D  0000E P.AAB:  .ASCII  <13>\KEY$V_INITIDX\           ;
```

```
     53  59  45  4B  50  55  44  5F  56  24  59  45  4B  0D  0001C P.AAC:   .ASCII   <13>\KEY$V_DUPKEYS\
     53  59  45  4B  47  48  43  5F  56  24  59  45  4B  0D  0002A P.AAD:   .ASCII   <13>\KEY$V_CHGKEYS\
     53  59  45  4B  4C  55  4E  5F  56  24  59  45  4B  0D  00038 P.AAE:   .ASCII   <13>\KEY$V_NULKEYS\
     58  44  49  54  49  4E  49  5F  56  24  59  45  4B  0D  00046 P.AAF:   .ASCII   <13>\KEY$V_INITIDX\
     53  59  45  4B  50  55  44  5F  56  24  59  45  4B  0D  00054 P.AAG:   .ASCII   <13>\KEY$V_DUPKEYS\
 50  4D  4F  43  5F  58  44  49  5F  56  24  59  45  4B  0F  00062 P.AAH:   .ASCII   <15>\KEY$V_IDX_COMPR\
                                                         52  00071
     58  44  49  54  49  4E  49  5F  56  24  59  45  4B  0D  00072 P.AAI:   .ASCII   <13>\KEY$V_INITIDX\
 50  4D  4F  43  5F  59  45  4B  5F  56  24  59  45  4B  0F  00080 P.AAJ:   .ASCII   <15>\KEY$V_KEY_COMPR\
                                                         52  0008F
 50  4D  4F  43  5F  43  45  52  5F  56  24  59  45  4B  0F  00090 P.AAK:   .ASCII   <15>\KEY$V_REC_COMPR\
                                                         52  0009F
     53  59  45  4B  50  55  44  5F  56  24  59  45  4B  0D  000A0 P.AAL:   .ASCII   <13>\KEY$V_DUPKEYS\
     53  59  45  4B  47  48  43  5F  56  24  59  45  4B  0D  000AE P.AAM:   .ASCII   <13>\KEY$V_CHGKEYS\
     53  59  45  4B  4C  55  4E  5F  56  24  59  45  4B  0D  000BC P.AAN:   .ASCII   <13>\KEY$V_NULKEYS\
 50  4D  4F  43  5F  58  44  49  5F  56  24  59  45  4B  0F  000CA P.AAO:   .ASCII   <15>\KEY$V_IDX_COMPR\
                                                         52  000D9
     58  44  49  54  49  4E  49  5F  56  24  59  45  4B  0D  000DA P.AAP:   .ASCII   <13>\KEY$V_INITIDX\
 50  4D  4F  43  5F  59  45  4B  5F  56  24  59  45  4B  0F  000E8 P.AAQ:   .ASCII   <15>\KEY$V_KEY_COMPR\
                                                         52  000F7
                                 67  6E  69  72  74  73  06  000F8 P.AAR:   .ASCII   <6>\string\
             64  72  6F  77  20  64  65  6E  67  69  73  0B  000FF P.AAS:   .ASCII   <11>\signed word\
     64  72  6F  77  20  64  65  6E  67  69  73  6E  75  0D  0010B P.AAT:   .ASCII   <13>\unsigned word\
 72  6F  77  67  6E  6F  6C  20  64  65  6E  67  69  73  0F  00119 P.AAU:   .ASCII   <15>\signed longword\
                                                         64  00128
 77  67  6E  6F  6C  20  64  65  6E  67  69  73  6E  75  11  00129 P.AAV:   .ASCII   <17>\unsigned longword\
                                                 64  72  6F  00138
 6C  61  6D  69  63  65  64  20  64  65  6B  63  61  70  0E  0013B P.AAW:   .ASCII   <14>\packed decimal\
 72  6F  77  64  61  75  71  20  64  65  6E  67  69  73  0F  0014A P.AAX:   .ASCII   <15>\signed quadword\
                                                         64  00159
 77  64  61  75  71  20  64  65  6E  67  69  73  6E  75  11  0015A P.AAY:   .ASCII   <17>\unsigned quadword\
                                                 64  72  6F  00169
                                             3F  3F  3F  03  0016C P.AAZ:   .ASCII   <3>\???\

                                                                            .PSECT   $OWN$,NOEXE,2

                                                 00000004  00000 KEY2_PRIMARY_DEF:
                                                                            .LONG    4
                                                 00000000' 00004            .ADDRESS P.AAA
                             00000000  00000000  00000000  00008            .LONG    0, 0, 0
                                                 00000000' 00014            .ADDRESS P.AAB
                                                 00000004  00018 KEY2_SECONDARY_DEF:
                                                                            .LONG    4
                             00000000' 00000000' 00000000' 0001C            .ADDRESS P.AAC, P.AAD, P.AAE
                                                 00000000  00028            .LONG    0
                                                 00000000' 0002C            .ADDRESS P.AAF
                                                 00000007  00030 KEY3_PRIMARY_DEF:
                                                                            .LONG    7
                                                 00000000' 00034            .ADDRESS P.AAG
                                       00000000  0000000C  00038            .LONG    0, 0
                                       00000000' 00000000' 00040            .ADDRESS P.AAH, P.AAI
                                                 00000000  00048            .LONG    0
                                       00000000' 00000000' 0004C            .ADDRESS P.AAJ, P.AAK
                                                 00000006  00054 KEY3_SECONDARY_DEF:
                                                                            .LONG    6
         00000000' 00000000' 00000000' 00000000' 00000000' 00058            .ADDRESS P.AAL, P.AAM, P.AAN, P.AAO, P.AAP
                                                 00000000  0006C            .LONG    0
```

```
                                                  00000000' 00070               .ADDRESS P.AAQ                                      ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00074 DATA_TYPES:
                                                                                .ADDRESS P.AAR, P.AAS, P.AAT, P.AAU, P.AAV, -       ;
                                        00000000' 00000000' 0008C                       P.AAW, P.AAX, P.AAY                         ;


                                                                                .PSECT  $CODE$,NOWRT,2

                                                       0000 00000 DATA_TYPE_NAME:
                                                                                .WORD   Save nothing                                ; 0910
                                       50        04    AC  D0 00002             MOVL    CODE, R0                                    ; 0924
                                       07              50  D1 00006             CMPL    R0, #7                                      ;
                                                       07  1B 00009             BLEQU   1$                                          ;
                                       51      0000'   CF  9E 0000B             MOVAB   P.AAZ, R1                                   ;
                                                       06  11 00010             BRB     2$                                          ;
                                       51      0000'CF40   D0 00012 1$:         MOV_    DATA_TYPES[R0], R1                          ; 0925
                                       50              51  D0 00018 2$:         MOVL    R1, R0                                      ; 0924
                                                           04 0001B             RET                                                 ; 0926

; Routine Size:  28 bytes,    Routine Base:  $CODE$ + 01FE
;  427         0927  2 ! First thing we need to do is ensure that the key descriptor fits in the
;  428         0928  2 ! block.  If not, we complain and signal a drastic error.
;  429         0929  2
;  430         0930  2 sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
;  431         0931  3 if .sp+key$c_bln geqa .b[bsd$l_endptr] then (
;  432         0932  3         anl$format_error(anlrms$_badkeyfit,.b[bsd$l_vbn],.key_id);
;  433         0933  3         signal (anlrms$_unwind);
;  434         0934  2 );
```

```
2 ! Now we can format the key descriptor, if requested.
2
3 if .report then (
3
3         ! Begin with a heading, containing the key of reference number.
3
3         anl$format_line(3,.indent_level,anlrms$_idxkey,.sp[key$b_keyref],
3                 .b[bsd$l_vbn],.b[bsd$l_offset]);
3         anl$format_skip(0);
3
3         ! Now the next key VBN and offset, if present.
3
3         if .sp[key$l_idxfl] nequ 0 then
3                 anl$format_line(0,.indent_level+1,anlrms$_idxkeynext,
3                         .sp[key$l_idxfl],.sp[key$w_noff]);
3
3         ! Now the area IDs.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeyareas,.sp[key$b_ianum],.sp[key$b_lanum],.sp[key$b_da
3
3         ! Now the index root level number.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeyrootlvl,.sp[key$b_rootlev]);
3
3         ! Now the bucket sizes.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeybktsz,.sp[key$b_idxbktsz],.sp[key$b_datbktsz]);
3
3         ! Now the root bucket VBN, if present.
3
3         if not .sp[key$v_initidx] then
3                 anl$format_line(0,.indent_level+1,anlrms$_idxkeyrootvbn,.sp[key$l_rootvbn]);
3
3         ! Now the flags.
3
3         anl$format_flags(.indent_level+1,anlrms$_idxkeyflags,.sp[key$b_flags],
4                 (if .anl$gw_prolog eqlu plg$c_ver_3 then
4                         if .sp[key$b_keyref] eqlu 0 then key3_primary_def
4                                                  else key3_secondary_def
4                 else
4                         if .sp[key$b_keyref] eqlu 0 then key2_primary_def
4                                                  else key2_secondary_def
3                 ));
3
3         ! Now the number of key segments.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeysegs,.sp[key$b_segments]);
3
3         ! Now the null character, if enabled.
3
3         if .sp[key$v_nulkeys] then
3                 anl$format_line(0,.indent_level+1,anlrms$_idxkeynull,.sp[key$b_nullchar]);
3
3         ! Now the total key size.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeykeysz,.sp[key$b_keysz]);
3
9      ;  493               0992  3          ! Now the minimum record length.
       ;  494               0993  3
0      ;  495               0994  3          anl$format_line(0,.indent_level+1,anlrms$_idxkeyminrec,.sp[key$w_minrecsz]);
       ;  496               0995  3
1      ;  497               0996  3          ! Now the fill quantities.
       ;  498               0997  3
       ;  499               0998  3          anl$format_line(0,.indent_level+1,anlrms$_idxkeyfill,.sp[key$w_idxfill],.sp[key$w_datfill]);
5      ;  500               0999  3
       ;  501               1000  3          ! Now the segment positions and sizes.
       ;  502               1001  3
       ;  503               1002  3          anl$format_line(0,.indent_level+1,anlrms$_idxkeyposs,.sp[key$b_segments],
       ;  504               1003  3                  .sp[key$w_position0],   .sp[key$w_position1],
       ;  505               1004  3                  .sp[key$w_position2],   .sp[key$w_position3],
       ;  506               1005  3                  .sp[key$w_position4],   .sp[key$w_position5],
       ;  507               1006  3                  .sp[key$w_position6],   .sp[key$w_position7]);
       ;  508               1007  3          anl$format_line(0,.indent_level+1,anlrms$_idxkeysizes,.sp[key$b_segments],
       ;  509               1008  3                  .sp[key$b_size0],       .sp[key$b_size1],
       ;  510               1009  3                  .sp[key$b_size2],       .sp[key$b_size3],
       ;  511               1010  3                  .sp[key$b_size4],       .sp[key$b_size5],
       ;  512               1011  3                  .sp[key$b_size6],       .sp[key$b_size7]);
       ;  513               1012  3
       ;  514               1013  3          ! Now we need to format the data type of the key segment(s).
       ;  515               1014  3
       ;  516               1015  3          anl$format_line(0,.indent_level+1,anlrms$_idxkey1type,data_type_name(.sp[key$b_datatype]));
       ;  517               1016  3
       ;  518               1017  3          ! Now the key name.  We use PREPARE_QUOTED_STRING to remove trialing
       ;  519               1018  3          ! NULs and enclose the name in quotes.
       ;  520               1019  3
       ;  521               1020  4          begin
       ;  522               1021  4          local
       ;  523               1022  4                  name_dsc: descriptor,
       ;  524               1023  4                  local_described_buffer(string_buf,key$s_keynam*2+2);
       ;  525               1024  4
       ;  526               1025  4          build_descriptor(name_dsc, key$s_keynam,sp[key$t_keynam]);
       ;  527               1026  4          anl$prepare_quoted_string(name_dsc,string_buf);
       ;  528               1027  4          anl$format_line(0,.indent_level+1,anlrms$_idxkeyname,string_buf);
       ;  529               1028  3          end;
       ;  530               1029  3
       ;  531               1030  3          ! And finally, the first data bucket VBN, if present.
       ;  532               1031  3
       ;  533               1032  3          if not .sp[key$v_initidx] then
       ;  534               1033  3                  anl$format_line(0,.indent_level+1,anlrms$_idxkeydatavbn,.sp[key$l_ldvbn]);
       ;  535               1034  2 );
```

```
2 ! Now we are going to check the contents of the key descriptor.  This is
2 ! a fairly rigorous test, but doesn't check anything that requires looking
2 ! at other structures (except as passed in the areas vector).
2
2 ! Start by ensuring that the three area IDs represent defined areas.
2 ! This check can only be made if the areas vector was passed.
2
2 if not nullparameter(3) then
2         if .areas_vector[.sp[key$b_ianum]] eqlu 0 or
2            .areas_vector[.sp[key$b_lanum]] eqlu 0 or
2            .areas_vector[.sp[key$b_danum]] eqlu 0       then
2                 anl$format_error(anlrms$_badkeyareaid,.b[bsd$l_vbn],.key_id);
2
2 ! Make sure the root level is at least 1.  This check cannot be made
2 ! if the index is uninitialized.
2
2 if not .sp[key$v_initidx] and .sp[key$b_rootlev] eqlu 0 then
2         anl$format_error(anlrms$_badkeyrootlevel,.b[bsd$l_vbn],.key_id);
2
2 ! The following two checks can only be made if the areas vector was passed.
2
3 if not nullparameter(3) then (
3
3         ! The index bucket size must be correct, and the two index area IDs
3         ! must have the same bucket size.
3
3         if .sp[key$b_idxbktsz] nequ .areas_vector[.sp[key$b_ianum]] or
3            .sp[key$b_idxbktsz] nequ .areas_vector[.sp[key$b_lanum]]     then
3                 anl$format_error(anlrms$_badkeyidxbkt,.b[bsd$l_vbn],.key_id);
3
3         ! The data bucket size must be correct.
3
3         if .sp[key$b_datbktsz] nequ .areas_vector[.sp[key$b_danum]] then
3                 anl$format_error(anlrms$_badkeydatabkt,.b[bsd$l_vbn],.key_id);
2 );
2
2 ! Check the key flags.
2
2 anl$check_flags(.b[bsd$l_vbn],.sp[key$b_flags],
3                 (if .anl$gw_prolog eqlu plg$c_ver_3 then
3                         if .sp[key$b_keyref] eqlu 0 then key3_primary_def
3                                                     else key3_secondary_def
3                 else
3                         if .sp[key$b_keyref] eqlu 0 then key2_primary_def
3                                                     else key2_secondary_def
2                 ));
2
2 ! Check the data type of the key.
2
2 if .sp[key$b_datatype] gtru key$c_max_data then
2         anl$format_error(anlrms$_badkeydatatype,.b[bsd$l_vbn],.sp[key$b_datatype],.key_id);
2
2 ! Check the number of key segments.
2
2 if .sp[key$b_segments] eqlu 0 or
2    .sp[key$b_segments] gtru (if .sp[key$b_datatype] eqlu key$c_string then 8 else 1) then
2         anl$format_error(anlrms$_badkeysegcount,.b[bsd$l_vbn],.sp[key$b_segments],.key_id);
```

```
2
2 ! Now we are going to check the key segment information.  We sit in a loop
2 ! and calculate the total key length and the length of a record required
2 ! to hold the key.
2
3 begin
3 bind
3         position_vector = sp[key$w_position0]: vector[8,word],
3         size_vector = sp[key$b_size0]: vector[8,byte];
3
3 total_size = required_record = 0;
4 incru i from 0 to 7 do (
4
5         if .i lssu .sp[key$b_segments] then (
5                 total_size = .total_size + .size_vector[.i];
5                 required_record = maxu(.required_record,.position_vector[.i]+.size_vector[.i]);
5
4         ) else
4                 if .position_vector[.i] nequ 0 or .size_vector[.i] nequ 0 then
4                         anl$format_error(anlrms$_badkeysegvec,.b[bsd$l_vbn],.key_id);
3 );
2 end;
2
2 ! Now make sure that the calculated information agrees with the information
2 ! in the descriptor.
2
2 if .sp[key$b_keysz] nequ .total_size or
2    .sp[key$w_minrecsz] nequ .required_record then
2         anl$format_error(anlrms$_badkeysummary,.b[bsd$l_vbn],.key_id);
2
2 ! Check the key of reference ID.
2
2 if .sp[key$b_keyref] nequ .key_id then
2         anl$format_error(anlrms$_badkeyrefid,.b[bsd$l_vbn],.key_id);
2
2 ! Check the index and data fill quantities.
2
2 if .sp[key$w_idxfill] gtru .sp[key$b_idxbktsz]*512 or
2    .sp[key$w_datfill] gtru .sp[key$b_datbktsz]*512      then
2         anl$format_error(anlrms$_badkeyfill,.b[bsd$l_vbn],.key_id);
```

```
;  635     1132  2 ! Now we are going to move along to the next key descriptor, if there is
;  636     1133  2 ! one.  If not, let's just quit.
;  637     1134  2
;  638     1135  2 if .sp[key$l_idxfl] eqlu 0 then
;  639     1136  2         return false;
;  640     1137  2
;  641     1138  2 !  Update the BSD and get the next key descriptor.
;  642     1139  2
;  643     1140  2 b[bsd$l_vbn] = .sp[key$l_idxfl];
;  644     1141  2 b[bsd$l_offset] = .sp[key$w_noff];
;  645     1142  2 anl$bucket(b,0);
;  646     1143  2
;  647     1144  2 return true;
;  648     1145  2
;  649     1146  1 end;
```

```
                                 0FFC 00000          .ENTRY  ANL$KEY_DESCRIPTOR, Save R2,R3,R4,R5,R6,R7,-; 0849
                                                             R8,R9,R10,R11                                ;
                    5B      0000G   CF   9E 00002          MOVAB   ANL$FORMAT_LINE, R11                       ;
                    5E         AC   AE   9E 00007          MOVAB   -84(SP), SP                                ;
                    55         04   AC   D0 0000B          MOVL    THE_BSD, R5                                ; 0852
                    53         0C   AC   D0 0000F          MOVL    AREAS, R3                                  ; 0853
  52         0C     A5         08   A5   C1 00013          ADDL3   8(R5), 12(R5), SP                          ; 0930
                    51         60   A2   9E 00019          MOVAB   96(R2), R1                                 ; 0931
             10     A5              51   D1 0001D          CMPL    R1, 16(R5)                                 ;
                                    1E   1F 00021          BLSSU   1$                                         ;
                               08   AC   DD 00023          PUSHL   KEY_ID                                     ; 0932
                               04   A5   DD 00026          PUSHL   4(R5)                                      ;
                        00000000G   8F   DD 00029          PUSHL   #ANLRMS$_BADKEYFIT                         ;
          0000G     CF              03   FB 0002F          CALLS   #3, ANL$FORMAT_ERROR                       ;
                        00000000G   8F   DD 00034          PUSHL   #ANLRMS$_UNWIND                            ; 0933
      00000000G     00              01   FB 0003A          CALLS   #1, LIB$SIGNAL                             ;
                    03         10   AC   E8 00041 1$:      BLBS    REPORT, 2$                                 ; 0937
                                  01E6   31 00045          BRW     10$                                        ;
                    7E         04   A5   7D 00048 2$:      MOVQ    4(R5), -(SP)                               ; 0942
                    7E         15   A2   9A 0004C          MOVZBL  21(SP), -(SP)                              ; 0941
                        00000000G   8F   DD 00050          PUSHL   #ANLRMS$_IDXKEY                            ;
                               14   AC   DD 00056          PUSHL   INDENT_LEVEL                               ;
                                    03   DD 00059          PUSHL   #3                                         ;
                    6B              06   FB 0005B          CALLS   #6, ANL$FORMAT_LINE                        ;
                                    7E   D4 0005E          CLRL    -(SP)                                      ; 0943
          0000G     CF              01   FB 00060          CALLS   #1, ANL$FORMAT_SKIP                        ;
                                    62   D5 00065          TSTL    (SP)                                       ; 0947
                                    16   13 00067          BEQL    3$                                         ;
                    7E         04   A2   3C 00069          MOVZWL  4(SP), -(SP)                               ; 0949
                                    62   DD 0006D          PUSHL   (SP)                                       ;
                        00000000G   8F   DD 0006F          PUSHL   #ANLRMS$_IDXKEYNEXT                        ; 0948
  7E         14     AC              01   C1 00075          ADDL3   #1, INDENT_LEVEL, -(SP)                    ;
                                    7E   D4 0007A          CLRL    -(SP)                                      ;
                    6B              05   FB 0007C          CALLS   #5, ANL$FORMAT_LINE                        ;
                    7E         08   A2   9A 0007F 3$:      MOVZBL  8(SP), -(SP)                               ; 0953
                    7E         07   A2   9A 00083          MOVZBL  7(SP), -(SP)                               ;
                    7E         06   A2   9A 00087          MOVZBL  6(SP), -(SP)                               ;
```

```
                          00000000G  8F  DD 0008B        PUSHL   #ANLRMS$_IDXKEYAREAS                    :
     54       14     AC              01  C1 00091        ADDL3   #1, INDENT_LEVEL, R4                    :
                                     54  DD 00096        PUSHL   R4                                      :
                                     7E  D4 00G98        CLRL    -(SP)                                   :
                     6B              06  FB 0009A        CALLS   #6, ANL$FORMAT_LINE                     :
                     7E         09   A2  9A 0009D        MOVZBL  9(SP), -(SP)                            : 0957
                          00000000G  8F  DD 000A1        PUSHL   #ANLRMS$_IDXKEYROOTLVL                  :
                                     54  DD 000A7        PUSHL   R4                                      :
                                     7E  D4 000A9        CLRL    -(SP)                                   :
                     6B              04  FB 000AB        CALLS   #4, ANL$FORMAT_LINE                     :
                     7E         0B   A2  9A 000AE        MOVZBL  11(SP), -(SP)                           : 0961
                     7E         0A   A2  9A 000B2        MOVZBL  10(SP), -(SP)                           :
                          00000000G  8F  DD 000B6        PUSHL   #ANLRMS$_IDXKEYBKTSZ                    :
                                     54  DD 000BC        PUSHL   R4                                      :
                                     7E  D4 000BE        CLRL    -(SP)                                   :
                     6B              J5  FB 000C0        CALLS   #5, ANL$FORMAT_LINF                     :
     10       10     A2              04  E0 000C3        BBS     #4, 16(SP), 4$                          : 0965
                                0C   A2  DD 000C8        PUSHL   12(SP)                                  : 0966
                          00000000G  8F  DD 000CB        PUSHL   #ANLRMS$_IDXKEYROOTVBN                  :
                                     54  DD 000D1        PUSHL   R4                                      :
                                     7E  D4 000D3        CLRL    -(SP)                                   :
                     6B              04  FB 000D5        CALLS   #4, ANL$FORMAT_LINE                     :
                     03      0000G   CF  B1 000D8 4$:    CMPW    ANL$GW_PROLOG, #3                       : 0971
                                     13  12 000DD        BNEQ    6$                                      :
                                15   A2  95 000DF        TSTB    21(SP)                                  : 0972
                                     07  12 000E2        BNEQ    5$                                      :
                     50      0000'   CF  9E 000E4        MOVAB   KEY3_PRIMARY_DEF, R0                    :
                                     18  11 000E9        BRB     8$                                      :
                     50      0000'   CF  9E 000EB 5$:    MOVAB   KEY3_SECONDARY_DEF, R0                  :
                                     11  11 000F0        BRB     8$                                      :
                                15   A2  95 000F2 6$:    TSTB    21(SP)                                  : 0975
                                     07  12 000F5        BNEQ    7$                                      :
                     50      0000'   CF  9E 000F7        MOVAB   KEY2_PRIMARY_DEF, R0                    :
                                     05  11 000FC        BRB     8$                                      :
                     50      0000'   CF  9E 000FE 7$:    MOVAB   KEY2_SECONDARY_DEF, R0                  :
                                     50  DD 00103 8$:    PUSHL   R0                                      :
                     7E         10   A2  9A 00105        MOVZBL  16(SP), -(SP)                           : 0970
                          00000000G  8F  DD 00109        PUSHL   #ANLRMS$_IDXKEYFLAGS                    :
                                     54  DD 0010F        PUSHL   R4                                      :
              0000G  CF              04  FB 00111        CALLS   #4, ANL$FORMAT_FLAGS                    :
                     7E         12   A2  9A 00116        MOVZBL  18(SP), -(SP)                           : 0981
                          00000000G  8F  DD 0011A        PUSHL   #ANLRMS$_IDXKEYSEGS                     :
                                     54  DD 00120        PUSHL   R4                                      :
                                     7E  D4 00122        CLRL    -(SP)                                   :
                     6B              04  FB 00124        CALLS   #4, ANL$FORMAT_LINE                     :
     11       10     A2              02  E1 00127        BBC     #2, 16(SP), 9$                          : 0985
                     7E         13   A2  9A 0012C        MOVZBL  19(SP), -(SP)                           : 0986
                          00000000G  8F  DD 00130        PUSHL   #ANLRMS$_IDXKEYNULL                     :
                                     54  DD 00136        PUSHL   R4                                      :
                                     7E  D4 00138        CLRL    -(SP)                                   :
                     6B              04  FB 0013A        CALLS   #4, ANL$FORMAT_LINE                     :
                     7E         14   A2  9A 0013D 9$:    MOVZBL  20(SP), -(SP)                           : 0990
                          00000000G  8F  DD 00141        PUSHL   #ANLRMS$_IDXKEYKEYSZ                    :
                                     54  DD 00147        PUSHL   R4                                      :
                                     7E  D4 00149        CLRL    -(SP)                                   :
                     6B              04  FB 0014B        CALLS   #4, ANL$FORMAT_LINE                     :
                     7E         16   A2  3C 0014E        MOVZWL  22(SP), -(SP)                           : 0994
```

```
                              00000000G  8F  DD 00152         PUSHL    #ANLRMS$_IDXKEYMINREC                  :
                                         54  DD 00158         PUSHL    R4                                     :
1                                        7E  D4 0015A         CLRL     -(SP)                                  :
                          6B             04  FB 0015C         CALLS    #4, ANL$FORMAT_LINE                    :
                          7E         1A  A2  3C 0015F         MOVZWL   26(SP), -(SP)                          : 0998
                          7E         18  A2  3C 00163         MOVZWL   24(SP), -(SP)                          :
3                             00000000G  8F  DD 00167         PUSHL    #ANLRMS$_IDXKEYFILL                    :
2                                        54  DD 0016D         PUSHL    R4                                     :
                                         7E  D4 0016F         CLRL     -(SP)                                  :
                          6B             05  FB 00171         CALLS    #5, ANL$FORMAT_LINE                    :
                          7E         2A  A2  3C 00174         MOVZWL   42(SP), -(SP)                          : 1006
7                         7E         28  A2  3C 00178         MOVZWL   40(SP), -(SP)                          :
                          7E         26  A2  3C 0017C         MOVZWL   38(SP), -(SP)                          : 1005
                          7E         24  A2  3C 00180         MOVZWL   36(SP), -(SP)                          :
                          7E         22  A2  3C 00184         MOVZWL   34(SP), -(SP)                          : 1004
                          7E         20  A2  3C 00188         MOVZWL   32(SP), -(SP)                          :
4                         7E         1E  A2  3C 0018C         MOVZWL   30(SP), -(SP)                          : 1003
                          7E         1C  A2  3C 00190         MOVZWL   28(SP), -(SP)                          :
                          7E         12  A2  9A 00194         MOVZBL   18(SP), -(SP)                          : 1002
                              00000000G  8F  DD 00198         PUSHL    #ANLRMS$_IDXKEYPOSS                    :
5                                        54  DD 0019E         PUSHL    R4                                     :
                                         7E  D4 001A0         CLRL     -(SP)                                  :
                          6B             0C  FB 001A2         CALLS    #12, ANL$FORMAT_LINE                   :
                          7E         33  A2  9A 001A5         MOVZBL   51(SP), -(SP)                          : 1011
                          7E         32  A2  9A 001A9         MOVZBL   50(SP), -(SP)                          :
7                         7E         31  A2  9A 001AD         MOVZBL   49(SP), -(SP)                          : 1010
                          7E         30  A2  9A 001B1         MOVZBL   48(SP), -(SP)                          :
                          7E         2F  A2  9A 001B5         MOVZBL   47(SP), -(SP)                          : 1009
                          7E         2E  A2  9A 001B9         MOVZBL   46(SP), -(SP)                          :
                          7E         2D  A2  9A 001BD         MOVZBL   45(SP), -(SP)                          : 1008
5                         7E         2C  A2  9A 001C1         MOVZBL   44(SP), -(SP)                          :
                          7E         12  A2  9A 001C5         MOVZBL   18(SP), -(SP)                          : 1007
6                             00000000G  8F  DD 001C9         PUSHL    #ANLRMS$_IDXKEYSIZES                   :
                                         54  DD 001CF         PUSHL    R4                                     :
                                         7E  D4 001D1         CLRL     -(SP)                                  :
                          6B             0C  FB 001D3         CALLS    #12, ANL$FORMAT_LINE                   :
0                         7E         11  A2  9A 001D6         MOVZBL   17(SP), -(SP)                          : 1015
                   FE05   CF             01  FB 001DA         CALLS    #1, DATA_TYPE_NAME                     :
                                         50  DD 001DF         PUSHL    R0                                     :
1                             00000000G  8F  DD 001E1         PUSHL    #ANLRMS$_IDXKEY1TYPE                   :
                                         54  DD 001E7         PUSHL    R4                                     :
                                         7E  D4 001E9         CLRL     -(SP)                                  :
                          6B             04  FB 001EB         CALLS    #4, ANL$FORMAT_LINF                    :
                          6E         42  8F  9A 001EE         MOVZBL   #66, STRING_BUF                        : 1023
5                  04     AE         08  AE  9E 001F2         MOVAB    STRING_BUF+8, STRING_BUF+4             :
                   4C     AE             20  D0 001F7         MOVL     #32, NAME_DSC                          : 1025
                   50     AE         34  A2  9E 001FB         MOVAB    52(R2), NAME_DSC+4                     :
                                         5E  DD 00200         PUSHL    SP                                     : 1026
6                                    50  AE  9F 00202         PUSHAB   NAME_DSC                               :
                  0000G   CF             02  FB 00205         CALLS    #2, ANL$PREPARE_QUOTED_STRING          :
                                         5E  DD 0020A         PUSHL    SP                                     : 1027
                              00000000G  8F  DD 0020C         PUSHL    #ANLRMS$_IDXKEYNAME                    :
                                         54  DD 00212         PUSHL    R4                                     :
8                                        7E  D4 00214         CLRL     -(SP)                                  :
                          6B             04  FB 00216         CALLS    #4, ANL$FORMAT_LINE                    :
            10     10     A2             04  E0 00219         BBS      #4, 16(SP), 10$                        : 1032
                                     54  A2  DD 0021E         PUSHL    84(SP)                                 : 1033
```

```
                     00000000G    8F  DD 00221          PUSHL   #ANLRMS$_IDXKEYDATAVBN                  ;
                                  54  DD 00227          PUSHL   R4                                      ;
                                  7E  D4 00229          CLRL    -(SP)                                   ;
                 6B               04  FB 0022B          CALLS   #4, ANL$FORMAT_LINE                     ;
                 03               6C  91 0022E  10$:    CMPB    (AP), #3                                ; 1042
                                  31  1F 00231          BLSSU   12$                                     ;
                            0C    AC  D5 00233          TSTL    12(AP)                                  ;
                                  2C  13 00236          BEQL    12$                                     ;
                 50         06    A2  9A 00238          MOVZBL  6(SP), R0                               ; 1043
                                6043  95 0023C          TSTB    (R0)[R3]                                ;
                                  12  13 0023F          BEQL    11$                                     ;
                 50         07    A2  9A 00241          MOVZBL  7(SP), R0                               ; 1044
                                6043  95 00245          TSTB    (R0)[R3]                                ;
                                  09  13 00248          BEQL    11$                                     ;
                 50         08    A2  9A 0024A          MOVZBL  8(SP), R0                               ; 1045
                                6043  95 0024E          TSTB    (R0)[R3]                                ;
                                  11  12 00251          BNEQ    12$                                     ;
                            08    AC  DD 00253  11$:    PUSHL   KEY_ID                                  ; 1046
                            04    A5  DD 00256          PUSHL   4(R5)                                   ;
                     00000000G    8F  DD 00259          PUSHL   #ANLRMS$_BADKEYAREAID                   ;
          0000G  CF               03  FB 0025F          CALLS   #3, ANL$FORMAT_ERROR                    ;
      16     10  A2               04  E0 00264  12$:    BBS     #4, 16(SP), 13$                         ; 1051
                            09    A2  95 00269          TSTB    9(SP)                                   ;
                                  11  12 0026C          BNEQ    13$                                     ;
                            08    AC  DD 0026E          PUSHL   KEY_ID                                  ; 1052
                            04    A5  DD 00271          PUSHL   4(R5)                                   ;
                     00000000G    8F  DD 00274          PUSHL   #ANLRMS$_BADKEYROOTLEVEL                ;
          0000G  CF               03  FB 0027A          CALLS   #3, ANL$FORMAT_ERROR                    ;
                 03               6C  91 0027F  13$:    CMPB    (AP), #3                                ; 1056
                                  48  1F 00282          BLSSU   16$                                     ;
                            0C    AC  D5 00284          TSTL    12(AP)                                  ;
                                  43  13 00287          BEQL    16$                                     ;
                 50         06    A2  9A 00289          MOVZBL  6(SP), R0                               ; 1061
               6043         0A    A2  91 0028D          CMPB    10(SP), (R0)[R3]                        ;
                                  0B  12 00292          BNEQ    14$                                     ;
                 50         07    A2  9A 00294          MOVZBL  7(SP), R0                               ; 1062
               6043         0A    A2  91 00298          CMPB    10(SP), (R0)[R3]                        ;
                                  11  13 0029D          BEQL    15$                                     ;
                            08    AC  DD 0029F  14$:    PUSHL   KEY_ID                                  ; 1063
                            04    A5  DD 002A2          PUSHL   4(R5)                                   ;
                     00000000G    8F  DD 002A5          PUSHL   #ANLRMS$_BADKEYIDXBKT                   ;
          0000G  CF               03  FB 002AB          CALLS   #3, ANL$FORMAT_ERROR                    ;
                 50         08    A2  9A 002B0  15$:    MOVZBL  8(SP), R0                               ; 1067
               6043         0B    A2  91 002B4          CMPB    11(SP), (R0)[R3]                        ;
                                  11  13 002B9          BEQL    16$                                     ;
                            08    AC  DD 002BB          PUSHL   KEY_ID                                  ; 1068
                            04    A5  DD 002BE          PUSHL   4(R5)                                   ;
                     00000000G    8F  DD 002C1          PUSHL   #ANLRMS$_BADKEYDATABKT                  ;
          0000G  CF               03  FB 002C7          CALLS   #3, ANL$FORMAT_ERROR                    ;
                 03      0000G    CF  B1 002CC  16$:    CMPW    ANL$GW_PROLOG, #3                       ; 1074
                                  14  12 002D1          BNEQ    18$                                     ;
                 58         15    A2  9A 002D3          MOVZBL  21(SP), R8                              ; 1075
                                  07  12 002D7          BNEQ    17$                                     ;
                 50      0000'    CF  9E 002D9          MOVAB   KEY3_PRIMARY_DEF, R0                    ;
                                  19  11 002DE          BRB     20$                                     ;
                 50      0000'    CF  9E 002E0  17$:    MOVAB   KEY3_SECONDARY_DEF, R0                  ;
                                  12  11 002E5          BRB     20$                                     ;
```

```
            58         15   A2  9A 002E7 18$:   MOVZBL  21(SP), R8                          ; 1078
                            07  12 002EB        BNEQ    19$                                 ;
            50      0000'   CF  9E 002ED        MOVAB   KEY2_PRIMARY_DEF, R0                ;
                            05  11 002F2        BRB     20$                                 ;
            50      0000'   CF  9E 002F4 19$:   MOVAB   KEY2_SECONDARY_DEF, R0              ;
                            50  DD 002F9 20$:   PUSHL   R0                                  ;
            7E         10   A2  9A 002FB        MOVZBL  16(SP), -(SP)                       ; 1073
            56         04   A5  D0 002FF        MOVL    4(R5), R6                           ;
                            56  DD 00303        PUSHL   R6                                  ;
     0000G  CF              03  FB 00305        CALLS   #3, ANL$CHECK_FLAGS                 ;
            07         11   A2  91 0030A        CMPB    17(SP), #7                          ; 1084
                            14  1B 0030E        BLEQU   21$                                 ;
                       08   AC  DD 00310        PUSHL   KEY_ID                              ; 1085
            7E         11   A2  9A 00313        MOVZBL  17(SP), -(SP)                       ;
                            56  DD 00317        PUSHL   R6                                  ;
                00000000G   8F  DD 00319        PUSHL   #ANLRMS$_BADKEYDATATYPE             ;
     0000G  CF              04  FB 0031F        CALLS   #4, ANL$FORMAT_ERROR                ;
            57         12   A2  9A 00324 21$:   MOVZBL  18(SP), R7                          ; 1089
                            12  13 00328        BEQL    24$                                 ;
                       11   A2  95 0032A        TSTB    17(SP)                              ; 1090
                            05  12 0032D        BNEQ    22$                                 ;
            50              08  D0 0032F        MOVL    #8, R0                              ;
                            03  11 00332        BRB     23$                                 ;
            50              01  D0 00334 22$:   MOVL    #1, R0                              ;
            50              57  D1 00337 23$:   CMPL    R7, R0                              ;
                            11  1B 0033A        BLEQU   25$                                 ;
                       08   AC  DD 0033C 24$:   PUSHL   KEY_ID                              ; 1091
            7E              56  7D 0033F        MOVQ    R6, -(SP)                           ;
                00000000G   8F  DD 00342        PUSHL   #ANLRMS$_BADKEYSEGCOUNT             ;
     0000G  CF              04  FB 00348        CALLS   #4, ANL$FORMAT_ERROR                ;
                            59  D4 0034D 25$:   CLRL    TOTAL_SIZE                          ; 1102
                            53  7C 0034F        CLRQ    I                                   ; 1103
50          53              01  78 00351 26$:   ASHL    #1, I, R0                           ; 1107
            57              53  D1 00355        CMPL    I, R7                               ; 1105
                            27  1E 00358        BGEQU   28$                                 ;
            51         2C A243  9A 0035A        MOVZBL  44(SP)[I], R1                       ; 1106
            59              51  C0 0035F        ADDL2   R1, TOTAL_SIZE                      ;
                       1C A240  9F 00362        PUSHAB  28(SP)[R0]                          ; 1107
            51              9E  3C 00366        MOVZWL  @(SP)+, R1                          ;
            5A         2C A243  9A 00369        MOVZBL  44(SP)[I], R10                      ;
            51              5A  C0 0036E        ADDL2   R10, R1                             ;
            50              54  D0 00371        MOVL    REQUIRED_RECORD, R0                 ;
            51              50  D1 00374        CMPL    R0, R1                              ;
                            03  1E 00377        BGEQU   27$                                 ;
            50              51  D0 00379        MOVL    R1, R0                              ;
            54              50  D0 0037C 27$:   MOVL    R0, REQUIRED_RECORD                 ;
                            1E  11 0037F        BRB     30$                                 ; 1105
                       1C A240  9F 00381 28$:   PUSHAB  28(SP)[R0]                          ; 1110
                            9E  B5 00385        TSTW    @(SP)+                              ;
                            06  12 00387        BNEQ    29$                                 ;
                       2C A243  95 00389        TSTB    44(SP)[I]                           ;
                            10  13 0038D        BEQL    30$                                 ;
                       08   AC  DD 0038F 29$:   PUSHL   KEY_ID                              ; 1111
                            56  DD 00392        PUSHL   R6                                  ;
                00000000G   8F  DD 00394        PUSHL   #ANLRMS$_BADKEYSEGVEC               ;
     0000G  CF              03  FB 0039A        CALLS   #3, ANL$FORMAT_ERROR                ;
                            53  D6 0039F 30$:   INCL    I                                   ; 1103
```

```
                                          07          53  D1 003A1          CMPL     I, #7                                    :
                                                      AB  1B 003A4          BLEQU    26$                                      :
                    59       14    A2     08          00  ED 003A6          CMP.V    #0, #8, 20(SP), TOTAL_SIZE               : 1118
                                                      08  12 003AC          BN.Q     31$                                      :
                    54       16    A2     10          00  ED 003AE          CMPZV    #0, #16, 22(SP), REQUIRED_RECORD         : 1119
                                                      10  13 003B4          BEQL     32$                                      :
                                               08     AC  DD 003B6 31$:     PUSHL    KEY_ID                                   : 1120
                                                      56  DD 003B9          PUSHL    R6                                       :
                                        00000000G     8F  DD 003BB          PUSHL    #ANLRMS$_BADKEYSUMMARY                   :
                                  0000G   CF          03  FB 003C1          CALLS    #3, ANL$FORMAT_ERROR                     :
                                     08   AC          58  D1 003C6 32$:     CMPL     R8, KEY_ID                               : 1124
                                                      10  13 003CA          BEQL     33$                                      :
                                               08     AC  DD 003CC          PUSHL    KEY_ID                                   : 1125
                                                      56  DD 003CF          PUSHL    R6                                       :
                                        00000000G     8F  DD 003D1          PUSHL    #ANLRMS$_BADKEYREFID                     :
                                  0000G   CF          03  FB 003D7          CALLS    #3, ANL$FORMAT_ERROR                     :
                                          51   0A     A2  9A 003DC 33$:     MOVZBL   10(SP), R1                               : 1129
                               51         51          09  78 003E0          ASHL     #9, R1, R1                               :
                    51       18    A2     10          00  ED 003E4          CMPZV    #0, #16, 24(SP), R1                      :
                                                      10  1A 003EA          BGTRU    34$                                      :
                                          51   0B     A2  9A 003EC          MOVZBL   11(SP), R1                               : 1130
                               51         51          09  78 003F0          ASHL     #9, R1, R1                               :
                    51       1A    A2     10          00  ED 003F4          CMPZV    #0, #16, 26(SP), R1                      :
                                                      10  1B 003FA          BLEQU    35$                                      :
                                               08     AC  DD 003FC 34$:     PUSHL    KEY_ID                                   : 1131
                                                      56  DD 003FF          PUSHL    R6                                       :
                                        00000000G     8F  DD 00401          PUSHL    #ANLRMS$_BADKEYFILL                      :
                                  0000G   CF          03  FB 00407          CALLS    #3, ANL$FORMAT_ERROR                     :
                                                      62  D5 0040C 35$:     TSTL     (SP)                                     : 1135
                                                      16  13 0040E          BEQL     36$                                      :
                                     04   A5          62  D0 00410          MOVL     (SP), 4(R5)                              : 1140
                                     08   A5   04     A2  3C 00414          MOVZWL   4(SP), 8(R5)                             : 1141
                                                      7E  D4 00419          CLRL     -(SP)                                    : 1142
                                                      55  DD 0041B          PUSHL    R5                                       :
                                  0000G   CF          02  FB 0041D          CALLS    #2, ANL$BUCKET                           :
                                          50          01  D0 00422          MOVL     #1, R0                                   : 1144
                                                          04 00425          RET                                               :
                                                      50  D4 00426 36$:     CLRL     R0                                       : 1146
                                                          04 00428          RET                                               :

; Routine Size:  1065 bytes,    Routine Base:  $CODE$ + 021A
```

```
1 %sbttl 'ANL$2BUCKET_HEADER - Print and Check a Bucket Header'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for printing and checking the contents
1 !       of the bucket header in prolog 2 indexed file buckets.
1 !
1 ! Formal Parameters:
1 !       the_bsd         The address of a BSD describing the complete bucket.
1 !                       We update it to the next bucket.
1 !       area_id         The alleged ID of the area containing this bucket.
1 !       level           The alleged level of this bucket.
1 !       report          A boolean, true if we are to print a report.
1 !       indent_level    The indentation level of the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       True if there is another bucket in this chain, false otherwise.
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$2bucket_header(the_bsd,area_id,level,report,indent_level) = begin
2
2 bind
2       b = .the_bsd: bsd;
2
2 own
2       control_flags_def: block[3,long] initial(
2                               1,
2                               uplit byte (%ascic 'BKT$V_LASTBKT'),
2                               uplit byte (%ascic 'BKT$V_ROOTBKT')
2                               );
2
2 local
2       sp: ref block[,byte];
2
2
2 ! We know the bucket header fits in the bucket.
2
2 ! Now we can format the header if requested.
2
2 sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
3 if .report then (
3
3       ! Start with a nice header, containing the VBN.
3
3       anl$format_line(3,.indent_level,anlrms$_bkt,.b[bsd$l_vbn]);
3       anl$format_skip(0);
3
3       ! Format the check character.
;  708          1204  3
;  709          1205  3         anl$format_line(0,.indent_level+1,anlrms$_bktcheck,.sp[bkt$b_checkchar]);
;  710          1206  3
;  711          1207  3         . Format the area number.
;  712          1208  3
;  713          1209  3         anl$format_line(0,.indent_level+1,anlrms$_bktarea,.sp[bkt$b_areano]);
;  714          1210  3
;  715          1211  3         ! Now the VBN address sample.
;  716          1212  3
;  717          1213  3         anl$format_line(0,.indent_level+1,anlrms$_bktsample,.sp[bkt$w_adrsample]);
;  718          1214  3
;  719          1215  3         ! Now the free space offset.
;  720          1216  3
;  721          1217  3         anl$format_line(0,.indent_level+1,anlrms$_bktfree,.sp[bkt$w_freespace]);
;  722          1218  3
;  723          1219  3         ! Now the available record ID range.
;  724          1220  3
;  725          1221  3         anl$format_line(0,.indent_level+1,anlrms$_bktrecid,.sp[bkt$b_nxtrecid],.sp[bkt$b_lstrecid]);
;  726          1222  3
;  727          1223  3         ! Now the next bucket VBN.
;  728          1224  3
;  729          1225  3         anl$format_line(0,.indent_level+1,anlrms$_bktnext,.sp[bkt$l_nxtbkt]);
;  730          1226  3
;  731          1227  3         ! Now the level number.
;  732          1228  3
;  733          1229  3         anl$format_line(0,.indent_level+1,anlrms$_bktlevel,.sp[bkt$b_level]);
;  734          1230  3
;  735          1231  3         ! And finally, the flags.
;  736          1232  3
;  737          1233  3         anl$format_flags(.indent_level+1,anlrms$_bktflags,.sp[bkt$b_bktcb],control_flags_def);
;  738          1234  2 );
;  740          1235  2 ! Now we are going the check the contents of the bucket header.  This is a
;  741          1236  2 ! fairly rigorous test, but doesn't check anything that requires looking
;  742          1237  2 ! at other structures.
;  743          1238  2
;  744          1239  2 ! Make sure the check byte is present in the last byte of the bucket.
;  745          1240  2
;  746          1241  2 if .sp[bkt$b_checkchar] nequ ch$rchar(.b[bsd$l_endptr]-1) then
;  747          1242  2         anl$format_error(anlrms$_badbktcheck,.b[bsd$l_vbn]);
;  748          1243  2
;  749          1244  2 ! Check the area ID.
;  750          1245  2
;  751          1246  2 if .sp[bkt$b_areano] nequ .area_id then
;  752          1247  2         anl$format_error(anlrms$_badbktareaid,.b[bsd$l_vbn]);
;  753          1248  2
;  754          1249  2 ! Check the bucket address sample.
;  755          1250  2
;  756          1251  2 if .sp[bkt$w_adrsample] nequ (.b[bsd$l_vbn] and %x'0000ffff') then
;  757          1252  2         anl$format_error(anlrms$_badbktsample,.b[bsd$l_vbn]);
;  758          1253  2
;  759          1254  2 ! Check that the next available byte is within reasonable limits.
;  760          1255  2
;  761          1256  2 if .sp[bkt$w_freespace] lssu bkt$c_overhdsz or
;  762          1257  2    .sp[bkt$w_freespace] gtru .b[bsd$w_size]*512-1 then
;  763          1258  2         anl$format_error(anlrms$_badbktfree,.b[bsd$l_vbn]);
;  764          1259  2
;  765          1260  2 ! Check the level number.
;  766          1261  2
;  767          1262  2 if .sp[bkt$b_level] nequ .level then
;  768          1263  2         anl$format_error(anlrms$_badbktlevel,.b[bsd$l_vbn]);
;  769          1264  2
;  770          1265  2 ! Check the byte of control flags.
;  771          1266  2
;  772          1267  2 anl$check_flags(.b[bsd$l_vbn],.sp[bkt$b_bktcb],control_flags_def);
;  773          1268  2
;  774        P 1269  2 statistics_callback(
;  775        P 1270  2
;  776        P 1271  2         ! If we are accumulating statistics, then we have to call the
;  777        P 1272  2         ! bucket callback routine, telling it the level, bucket size,
;  778        P 1273  2         ! and fill amount.
;  779        P 1274  2
;  780        P 1275  2         anl$bucket_callback(.sp[bkt$b_level],
;  781        P 1276  2                             .b[bsd$w_size],
;  782        P 1277  2                             .sp[bkt$w_freespace] + 1);
;  783          1278  2 );
```

```
:  785          1279  2 ! If this is not the last bucket in this chain, then let's update the
:  786          1280  2 ! BSD to describe the next one.  Otherwise forget it.
:  787          1281  2
:  788          1282  3 if not .sp[bkt$v_lastbkt] then (
:  789          1283  3         b[bsd$l_vbn] = .sp[bkt$l_nxtbkt];
:  790          1284  3         anl$bucket(b,0);
:  791          1285  3         return true;
:  792          1286  2 ) else
:  793          1287  2         return false;
:  794          1288  2
:  795          1289  1 end;



                                                                           .PSECT  $PLIT$,NOWRT,NOEXE,2

  54  4B  42  54  53  41  4C  5F  56  24  54  4B  42  0D  00170 P.ABA:  .ASCII  <13>\BKT$V_LASTBKT\                          :
  54  4B  42  54  4F  4F  52  5F  56  24  54  4B  42  0D  0017E P.ABB:  .ASCII  <13>\BKT$V_ROOTBKT\                          :

                                                                           .PSECT  $OWN$,NOEXE,2

                                                 00000001  00094 CONTROL_FLAGS_DEF:
                                                                           .LONG   1
                                       00000000' 00000000' 00098           .ADDRESS P.ABA, P.ABB                           :



                                                                           .PSECT  $CODE$,NOWRT,2

                                                    007C 00000             .ENTRY  ANL$2BUCKET_HEADER, Save R2,R3,R4,R5,R6      : 1175
                         56        0000G   CF   9E 00002             MOVAB   ANL$FORMAT_ERROR, R6                          :
                         55        0000G   CF   9E 00007             MOVAB   ANL$FORMAT_LINE, R5                           :
                         53           04   AC   D0 0000C             MOVL    THE_BSD, R3                                   : 1178
          52        0C   A3           08   A3   C1 00010             ADDL3   8(R3), 12(R3), SP                             : 1195
                         03           10   AC   E8 00016             BLBS    REPORT, 1$                                    : 1196
                                         00AB   31 0001A             BRW     2$                                            :
                                      04   A3   DD 0001D 1$:         PUSHL   4(R3)                                         : 1200
                               00000000G   8F   DD 00020             PUSHL   #ANLRMS$_BKT                                  :
                                      14   AC   DD 00026             PUSHL   INDENT_LEVEL                                  :
                                           03   DD 00029             PUSHL   #3                                            :
                         65                04   FB 0002B             CALLS   #4, ANL$FORMAT_LINE                           :
                                           7E   D4 0002E             CLRL    -(SP)                                         : 1201
                 0000G   CF                01   FB 00030             CALLS   #1, ANL$FORMAT_SKIP                           :
                         7E                62   9A 00035             MOVZBL  (SP), -(SP)                                   : 1205
                               00000000G   8F   DD 00038             PUSHL   #ANLRMS$_BKTCHECK                             :
          54        14   AC                01   C1 0003E             ADDL3   #1, INDENT_LEVEL, R4                          :
                                           54   DD 00043             PUSHL   R4                                            :
                                           7E   D4 00045             CLRL    -(SP)                                         :
                         65                04   FB 00047             CALLS   #4, ANL$FORMAT_LINE                           :
                         7E           01   A2   9A 0004A             MOVZBL  1(SP), -(SP)                                  : 1209
                               00000000G   8F   DD 0004E             PUSHL   #ANLRMS$_BKTAREA                              :
                                           54   DD 00054             PUSHL   R4                                            :
                                           7E   D4 00056             CLRL    -(SP)                                         :
                         65                04   FB 00058             CALLS   #4, ANL$FORMAT_LINE                           :
                         7E           02   A2   3C 0005B             MOVZWL  2(SP), -(SP)                                  : 1213
                               00000000G   8F   DD 0005F             PUSHL   #ANLRMS$_BKTSAMPLE                            :
                                           54   DD 00065             PUSHL   R4                                            :
                                                7E  D4 00067        CLRL    -(SP)                                   ;
                                 65             04  FB 00069        CALLS   #4, ANL$FORMAT_LINE                     ;
                                 7E         04  A2  3C 0006C        MOVZWL  4(SP), -(SP)                            ; 1217
                                     00000000G  8F  DD 00070        PUSHL   #ANLRMS$_BKTFREE                        ;
                                                54  DD 00076        PUSHL   R4                                      ;
                                                7E  D4 00078        CLRL    -(SP)                                   ;
                                 65             04  FB 0007A        CALLS   #4, ANL$FORMAT_LINE                     ;
                                 7E         07  A2  9A 0007D        MOVZBL  7(SP), -(SP)                            ; 1221
                                 7E         06  A2  9A 00081        MOVZBL  6(SP), -(SP)                            ;
                                     00000000G  8F  DD 00085        PUSHL   #ANLRMS$_BKTRECID                       ;
                                                54  DD 0008B        PUSHL   R4                                      ;
                                                7E  D4 0008D        CLRL    -(SP)                                   ;
                                 65             05  FB 0008F        CALLS   #5, ANL$FORMAT_LINE                     ;
                                            08  A2  DD 00092        PUSHL   8(SP)                                   ; 1225
                                     00000000G  8F  DD 00095        PUSHL   #ANLRMS$_BKTNEXT                        ;
                                                54  DD 0009B        PUSHL   R4                                      ;
                                                7E  D4 0009D        CLRL    -(SP)                                   ;
                                 65             04  FB 0009F        CALLS   #4, ANL$FORMAT_LINE                     ;
                                 7E         0C  A2  9A 000A2        MOVZBL  12(SP), -(SP)                           ; 1229
                                     00000000G  8F  DD 000A6        PUSHL   #ANLRMS$_BKTLEVEL                       ;
                                                54  DD 000AC        PUSHL   R4                                      ;
                                                7E  D4 000AE        CLRL    -(SP)                                   ;
                                 65             04  FB 000B0        CALLS   #4, ANL$FORMAT_LINE                     ;
                                         0000'  CF  9F 000B3        PUSHAB  CONTROL_FLAGS_DEF                       ; 1233
                                 7E         0D  A2  9A 000B7        MOVZBL  13(SP), -(SP)                           ;
                                     00000000G  8F  DD 000BB        PUSHL   #ANLRMS$_BKTFLAGS                       ;
                                                54  DD 000C1        PUSHL   R4                                      ;
                            0000GCF             04  FB 000C3        CALLS   #4, ANL$FORMAT_FLAGS                    ;
                                 50         10  A3  D0 000C8 2$:    MOVL    16(R3), R0                              ; 1241
                              FF A0             62  91 000CC        CMPB    (SP), -1(R0)                            ;
                                                0C  13 000D0        BEQL    3$                                      ;
                                            04  A3  DD 000D2        PUSHL   4(R3)                                   ; 1242
                                     00000000G  8F  DD 000D5        PUSHL   #ANLRMS$_BADBKTCHECK                    ;
                                 66             02  FB 000DB        CALLS   #2, ANL$FORMAT_ERROR                    ;
08   AC       01   A2            08             00  ED 000DE 3$:    CMPZV   #0, #8, 1(SP), AREA_ID                  ; 1246
                                                0C  13 000E5        BEQL    4$                                      ;
                                            04  A3  DD 000E7        PUSHL   4(R3)                                   ; 1247
                                     00000000G  8F  DD 000EA        PUSHL   #ANLRMS$_BADBKTAREAID                   ;
                                 66             02  FB 000F0        CALLS   #2, ANL$FORMAT_ERROR                    ;
                                 54         04  A3  D0 000F3 4$:    MOVL    4(R3), R4                               ; 1251
                                 54         02  A2  B1 000F7        CMPW    2(SP), R4                               ;
                                                0B  13 000FB        BEQL    5$                                      ;
                                                54  DD 000FD        PUSHL   R4                                      ; 1252
                                     00000000G  8F  DD 000FF        PUSHL   #ANLRMS$_BADBKTSAMPLE                   ;
                                 66             02  FB 00105        CALLS   #2, ANL$FORMAT_ERROR                    ;
                                 0E         04  A2  B1 00108 5$:    CMPW    4(SP), #14                              ; 1256
                                                12  1F 0010C        BLSSU   6$                                      ;
                                 50         02  A3  3C 0010E        MOVZWL  2(R3), R0                               ; 1257
                   50            50             09  78 00112        ASHL    #9, R0, R0                              ;
                                                50  D7 00116        DECL    R0                                      ;
     50       04   A2            10             00  ED 00118        CMPZV   #0, #16, 4(SP), R0                      ;
                                                0B  1B 0011E        BLEQU   7$                                      ;
                                                54  DD 00120 6$:    PUSHL   R4                                      ; 1258
                                     00000000G  8F  DD 00122        PUSHL   #ANLRMS$_BADBKTFREE                     ;
                                 66             02  FB 00128        CALLS   #2, ANL$FORMAT_ERROR                    ;
0C   AC       0C   A2            08             00  ED 0012B 7$:    CMPZV   #0, #8, 12(SP), LEVEL                   ; 1262
                                                0B  13 00132        BEQL    8$                                      ;
```

```
                          54  DD 00134          PUSHL   R4                                   ; 1263
               00000000G  8F  DD 00136          PUSHL   #ANLRMS$_BADBKTLEVEL                 ;
          66              02  FB 0013C          CALLS   #2, ANL$FORMAT_ERROR                 ;
                   0000'  CF  9F 0013F  8$:     PUSHAB  CONTROL_FLAGS_DEF                    ; 1267
          7E          0D  A2  9A 00143          MOVZBL  13(SP), -(SP)                        ;
                          54  DD 00147          PUSHL   R4                                   ;
   0000G  CF              03  FB 00149          CALLS   #3, ANL$CHECK_FLAGS                  ;
          02       0000G  CF  91 0014E          CMPB    ANL$GB_MODE, #2                      ; 1278
                          07  13 00153          BEQL    9$                                   ;
          04       0000G  CF  91 00155          CMPB    ANL$GB_MODE, #4                      ;
                          13  12 0015A          BNEQ    10$                                  ;
          7E          04  A2  3C 0015C  9$:     MOVZWL  4(SP), -(SP)                         ;
                          6E  D6 00160          INCL    (SP)                                 ;
          7E          02  A3  3C 00162          MOVZWL  2(R3), -(SP)                         ;
          7E          0C  A2  9A 00166          MOVZBL  12(SP), -(SP)                        ;
   0000G  CF              03  FB 0016A          CALLS   #3, ANL$BUCKET_CALLBACK              ;
          12          0D  A2  E8 0016F  10$:    BLBS    13(SP), 11$                          ; 1282
     04   A3          08  A2  D0 00173          MOVL    8(SP), 4(R3)                         ; 1283
                          7E  D4 00178          CLRL    -(SP)                                ; 1284
                          53  DD 0017A          PUSHL   R3                                   ;
   0000G  CF              02  FB 0017C          CALLS   #2, ANL$BUCKET                       ;
          50              01  D0 00181          MOVL    #1, R0                               ; 1287
                              04 00184          RET                                          ;
                          50  D4 00185  11$:    CLRL    R0                                   ;
                              04 00187          RET                                          ; 1289

; Routine Size:  392 bytes,    Routine Base:  $CODE$ + 0643
```

```
1 %sbttl 'ANL$2INDEX_RECORD - Print & Check an Index Record'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for printing and checking the contents
1 !       of a prolog 2 index record.  An index record is the structure present
1 !       in the indices of an 'ndexed file.
1 !
1 ! Formal Parameters:
1 !       rec_bsd         Address of BSD describing the index record.
1 !       key_bsd         Address of BSD describing key descriptor for index.
1 !       report          A boolean, true if we are to print the record.
1 !       indent_level    Indentation level for the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       True if there is another index record in this bucket, false otherwise.
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$2index_record(rec_bsd,key_bsd,report,indent_level) = begin
2
2 bind
2       b = .rec_bsd: bsd,
2       k = .key_bsd: bsd,
2       kp = .k[bsd$l_bufptr] + .k[bsd$l_offset]: block[,byte];
2
2 local
2       hp: ref block[,byte],
2       sp: ref block[,byte],
2       length: long;
2
2
2 ! First we have to ensure that this index record really fits in the used
2 ! space of the bucket.  If not, we have a drastic structure error.
2 ! Begin by ensuring that the first byte fits.
2
2 hp = .b[bsd$l_bufptr];
2
3 if .b[bsd$l_offset] gequ .hp[bkt$w_freespac.  then (
3       anl$format_error(anlrms$_badidxrecfit,.b[bsd$l_vbn]);
3       signal (anlrms$_unwind);
2 );
2
2 ! Now calculate the total length of the index record.
2
2 sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
2 length = 1 +
3         (case .sp[irc$v_ptrsz] from 0 to 3 of set
3           [0]:  2;
;  854          1347  3            [1]:  3;
;  855          1348  3            [2]:  4;
;  856          1349  4            [3]:  (anl$format_error(anlrms$_badidxrecps,.b[bsd$l_vbn]);
;  857          1350  3                   signal (anlrms$_unwind););
;  858          1351  2          tes) +
;  859          1352  2          .kp[key$b_keysz];
;  860          1353  2
;  861          1354  2  ! Now make sure the entire index record can fit into the used space.
;  862          1355  2
;  863          1356  3  if .b[bsd$l_offset]+.length gtru .hp[bkt$w_freespace] then (
;  864          1357  3          anl$format_error(anlrms$_badidxrecfit,.b[bsd$l_vbn]);
;  865          1358  3          signal (anlrms$_unwind);
;  866          1359  2  );
```

```
2 ! Now we can format the index record if requested by the caller.
2
3 if .report then (
3
3         ! Begin with a header.
3
3         anl$format_line(3,.indent_level,anlrms$_idxrec,.b[bsd$l_vbn],.b[bsd$l_offset]);
3         anl$format_skip(0);
3
3         ! Now the bucket pointer and its length.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxrecptr,.sp[irc$v_ptrsz]+2,
4                                 (case .sp[irc$v_ptrsz] from 0 to 2 of set
4                                 [0]:    .sp[1,0,16,0];
4                                 [1]:    .sp[1,0,24,0];
4                                 [2]:    .sp[1,0,32,0];
3                                 tes));
3
3         ! Now the key value.  Dump it in hex with a heading.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxkeybytes);
4         begin
4         local
4                 key_dsc: descriptor;
4
4         build_descriptor(key_dsc,.kp[key$b_keysz],.sp + 1 + .sp[irc$v_ptrsz]+2);
4         anl$format_hex(.indent_level+2,key_dsc);
3         end;
2 );
```

```
;  898           1389  2 ! Now we can actually check the integrity of the index record.  Most of the
;  899           1390  2 ! work involves checking its fit in the bucket, which has already been done.
;  900           1391  2 ! We have a few things left, however.
;  901           1392  2
;  902           1393  2 ! Check the index record control bits.  There aren't any.
;  903           1394  2
;  904           1395  2 if .sp[irc$v_recordcb] nequ 0 then
;  905           1396  2         anl$format_error(anlrms$_badidxrecbits,.b[bsd$l_vbn]);
;  906           1397  2
;  907         P 1398  2 statistics_callback(
;  908         P 1399  2
;  909         P 1400  2         ! If we are accumulating statistics, then we have to call the
;  910         P 1401  2         ! index record callback routine, telling it the level and overall
;  911         P 1402  2         ! record length.
;  912         P 1403  2
;  913         P 1404  2         anl$index_callback(.hp[bkt$b_level],
;  914         P 1405  2                            .length,
;  915         P 1406  2                            0);
;  916           1407  2 );
;  917           1408  2
;  918           1409  2 ! Now we can advance to the next index record.  If there isn't another
;  919           1410  2 ! one, then just return without modifying the BSD. Otherwise update
;  920           1411  2 ! the BSD.
;  921           1412  2
;  922           1413  3 if .b[bsd$l_offset]+.length lssu .hp[bkt$w_freespace] then (
;  923           1414  3         b[bsd$l_offset] = .b[bsd$l_offset] + .length;
;  924           1415  3         return true;
;  925           1416  2 ) else
;  926           1417  2         return false;
;  927           1418  2
;  928           1419  1 end;
; INFO#212                L1:1350
; Null expression appears in value-required context




                                                           0FFC 00000          .ENTRY  ANL$2INDEX_RECORD, Save R2,R3,R4,R5,R6,R7,- ; 1317
                                                                                       R8,R9,R10,R11                               ;
                                       5B 00000000G    00  9E 00002          MOVAB   LIB$SIGNAL, R11                             ;
                                       5A 00000000G    8F  D0 00009          MOVL    #ANLRMS$_UNWIND, R10                        ;
                                       5E              08  C2 00010          SUBL2   #8, SP                                      ;
                                       53          04  AC  D0 00013          MOVL    REC_BSD, R3                                 ; 1320
                                       50          08  AC  D0 00017          MOVL    KEY_BSD, R0                                 ; 1321
                     55         0C     A0          08  A0  C1 0001B          ADDL3   8(R0), 12(R0), R5                           ; 1322
                                       56          0C  A3  D0 00021          MOVL    12(R3), HP                                  ; 1334
   08    A3     04   A6                10              00  ED 00025          CMPZV   #0, #16, 4(HP), 8(R3)                       ; 1336
                                                       13  1A 0002C          BGTRU   1$                                          ;
                                                   04  A3  DD 0002E          PUSHL   4(R3)                                       ; 1337
                                          00000000G    8F  DD 00031          PUSHL   #ANLRMS$_BADIDXRECFIT                       ;
                             0000G     CF              02  FB 00037          CALLS   #2, ANL$FORMAT_ERROR                        ;
                                                       5A  DD 0003C          PUSHL   R10                                         ; 1338
                                       6B              01  FB 0003E          CALLS   #1, LIB$SIGNAL                              ;
                     52         0C     A3          08  A3  C1 00041 1$:      ADDL3   8(R3), 12(R3), SP                           ; 1343
         54          62                02              00  EF 00047          EXTZV   #0, #2, (SP), R4                            ; 1345
                     03                00              54  CF 0004C          CASEL   R4, #0, #3                                  ;
```

```
           0017      0012     000D     0008    00050 2$:    .WORD   3$-2$,-                       ;
                                                                    4$-2$,-                       ;
                                                                    5$-2$,-                       ;
                                                                    6$-2$                         ;
                                50       02 D0 00058 3$:    MOVL    #2, R0                        ;
                                         1F 11 0005B        BRB     7$                            ;
                                50       03 D0 0005D 4$:    MOVL    #3, R0                        ;
                                         1A 11 00060        BRB     7$                            ;
                                50       04 C0 00062 5$:    MOVL    #4, R0                        ;
                                         15 11 00065        BRB     7$                            ;
                                     04  A3 DD 00067 6$:    PUSHL   4(R3)                         ; 1349
                              00000000G  8F DD 0006A        PUSHL   #ANLRMS$_BADIDXRECPS          ;
                         0000G  CF       02 FB 00070        CALLS   #2, ANL$FORMAT_ERROR          ;
                                         5A DD 00075        PUSHL   R10                           ; 1350
                                6B       01 FB 00077        CALLS   #1, LIB$SIGNAL                ;
                                         50 D4 0007A        CLRL    R0                            ; 1345
                                57   14  A5 9A 0007C 7$:    MOVZBL  20(R5), R7                    ; 1352
                                58  01 A740 9E 00080        MOVAB   1(R7)[R0], LENGTH             ; 1351
                       59       58   08  A3 C1 00085        ADDL3   8(R3), LENGTH, R9             ; 1356
             59    04  A6       10       00 ED 0008A        CMPZV   #0, #16, 4(HP), R9            ;
                                         13 1E 00090        BGEQU   8$                            ;
                                     04  A3 DD 00092        PUSHL   4(R3)                         ; 1357
                              00000000G  8F DD 00095        PUSHL   #ANLRMS$_BADIDXRECFIT         ;
                         0000G  CF       02 FB 0009B        CALLS   #2, ANL$FORMAT_ERROR          ;
                                         5A DD 000A0        PUSHL   R10                           ; 1358
                                6B       01 FB 000A2        CALLS   #1, LIB$SIGNAL                ;
                                71   0C  AC E9 000A5 8$:    BLBC    REPORT, 14$                   ; 1362
                                7E   04  A3 7D 000A9        MOVQ    4(R3), -(SP)                  ; 1366
                              00000000G  8F DD 000AD        PUSHL   #ANLRMS$_IDXREC               ;
                                     10  AC DD 000B3        PUSHL   INDENT_LEVEL                  ;
                                         03 DD 000B6        PUSHL   #3                            ;
                         0000G  CF       05 FB 000B8        CALLS   #5, ANL$FORMAT_LINE           ;
                                         7E D4 000BD        CLRL    -(SP)                         ; 1367
                         0000G  CF       01 FB 000BF        CALLS   #1, ANL$FORMAT_SKIP           ;
                       02       00       54 CF 000C4        CASEL   R4, #0, #2                    ; 1372
                     0014     000C     0006    000C8 9$:    .WORD   10$-9$,-                      ;
                                                                    11$-9$,-                      ;
                                                                    12$-9$                        ;
                                7E   01  A2 3C 000CE 10$:   MOVZWL  1(SP), -(SP)                  ; 1373
                                         0B 11 000D2        BRB     13$                           ;
             7E    01  A2       18       00 EF 000D4 11$:   EXTZV   #0, #24, 1(SP), -(SP)         ; 1374
                                         03 11 000DA        BRB     13$                           ;
                                     01  A2 DD 000DC 12$:   PUSHL   1(SP)                         ; 1375
                                     02  A4 9F 000DF 13$:   PUSHAB  2(R4)                         ; 1371
                              00000000G  8F DD 000E2        PUSHL   #ANLRMS$_IDXRECPTR            ;
                       55   10  AC       01 C1 000E8        ADDL3   #1, INDENT_LEVEL, R5          ;
                                         55 DD 000ED        PUSHL   R5                            ;
                                         7E D4 000EF        CLRL    -(SP)                         ;
                         0000G  CF       05 FB 000F1        CALLS   #5, ANL$FORMAT_LINE           ;
                              00000000G  8F DD 000F6        PUSHL   #ANLRMS$_IDXKEYBYTES          ; 1380
                                         55 DD 000FC        PUSHL   R5                            ;
                                         7E D4 000FE        CLRL    -(SP)                         ;
                         0000G  CF       03 FB 00100        CALLS   #3, ANL$FORMAT_LINE           ;
                                6E       57 D0 00105        MOVL    R7, KEY_DSC                   ; 1385
                            04  AE  03 A442 9E 00108        MOVAB   3(R4)[SP], KEY_DSC+4          ;
                                         5E DD 0010E        PUSHL   SP                            ; 1386
                       7E   10  AC       02 C1 00110        ADDL3   #2, INDENT_LEVEL, -(SP)       ;
```

```
                            0000G  CF            02  FB 00115        CALLS   #2, ANL$FORMAT_HEX                 :
                               FC  8F            62  93 0011A 14$:   BITB    (SP), #252                         : 1395
                                                 0E  13 0011E        BEQL    15$                                :
                                         04      A3  DD 00120        PUSHL   4(R3)                              : 1396
                                     00000000G   8F  DD 00123        PUSHL   #ANLRMS$_BADIDXRECBITS             :
                            0000G  CF            02  FB 00129        CALLS   #2, ANL$FORMAT_ERROR               :
                                   02     0000G  CF  91 0012E 15$:   CMPB    ANL$GB_MODE, #2                    : 1407
                                                 07  13 00133        BEQL    16$                                :
                                   04     0000G  CF  91 00135        CMPB    ANL$GB_MODE, #4                    :
                                                 0D  12 0013A        BNEQ    17$                                :
                                                 7E  D4 0013C 16$:   CLRL    -(SP)                              :
                                                 58  DD 0013E        PUSHL   LENGTH                             :
                                   7E        0C  A6  9A 00140        MOVZBL  12(HP), -(SP)                      :
                            0000G  CF            03  FB 00144        CALLS   #3, ANL$INDEX_CALLBACK             :
             59        04    A6    10            00  ED 00149 17$:   CMPZV   #0, #16, 4(HP), R9                 : 1413
                                                 08  1B 0014F        BLEQU   18$                                :
                              08   A3            58  C0 00151        ADDL2   LENGTH, 8(R3)                      : 1414
                                   50            01  D0 00155        MOVL    #1, R0                             : 1417
                                                     04 00158        RET                                        :
                                                 50  D4 00159 18$:   CLRL    R0                                 :
                                                     04 0015B        RET                                        : 1419

; Routine Size:  348 bytes,    Routine Base:  $CODE$ + 07CB
```

```
1 %sbttl 'ANL$2PRIMARY_DATA_RECORD - Print & Check A Primary Data Record'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for printing and checking the contents
1 !       of a prolog 2 primary data record.  Primary data records exist in
1 !       the data buckets of the primary index.  They can contain actual data
1 !       records or RRVs.
1 !
1 ! Formal Parameters:
1 !       rec_bsd         Address of BSD describing the data record.
1 !       key_bsd         Address of BSD describing key for this index.
1 !       report          A boolean, true if we are to print the record.
1 !       indent_level    Indentation level for the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       True if there is another data record in this bucket, false otherwise.
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$2primary_data_record(rec_bsd,key_bsd,report,indent_level) = begin
2
2 bind
2       b = .rec_bsd: bsd;
2
2 own
2       data_flags_def: vector[6,long] initial(
2                               4,
2                               0,
2                               0,
2                               uplit byte (%ascic 'IRC$V_DELETED'),
2                               uplit byte (%ascic 'IRC$V_RRV'),
2                               uplit byte (%ascic 'IRC$V_NOPTRSZ')
2                               );
2 local
2       hp: ref block[,byte],
2       sp: ref block[,byte],
2       rp: ref block[,byte],
2       data_length: long, length: long;
2
2
2 ! First we have to ensure that this data record fits in the used space
2 ! of the bucket.  If not, we have a drastic structure error.  Begin by
2 ! ensuring that the first byte fits.
2
2 hp = .b[bsd$l_bufptr];
2
3 if .b[bsd$l_offset] gequ .hp[bkt$w_freespace] then (
3       anl$format_error(anlrms$_baddatarecfit,.b[bsd$l_vbn]);
3               signal (anlrms$_unwind);
2       );
2
2       ! Now calculate the length of the record not including the actual data.
2       ! Set up a pointer RP to the cata record.
2
2       sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
2       length = 1 +
2                1 +
3                (if .sp[irc$v_noptrsz] then 0 else
4                       (case .sp[irc$v_ptrsz] from 0 to 3 of set
4                        [0]:    3;
4                        [1]:    4;
4                        [2]:    5;
5                        [3]:    (anl$format_error(anlrms$_baddatarecps,.b[bsd$l_vbn]);
4                                signal (anlrms$_unwind););
4                       tes)
2                );
2       rp = .sp + .length;
2       if not .sp[irc$v_rrv] and .anl$gl_fat[fat$v_rtype] nequ fat$c_fixed then
2               length = .length + 2;
2
2       ! Now make sure that all those bytes fit into the used portion of the bucket.
2
3       if .b[bsd$l_offset]+.length gtru .hp[bkt$w_freespace] then (
3               anl$format_error(anlrms$_baddatarecfit,.b[bsd$l_vbn]);
3               signal (anlrms$_unwind);
2       );
2
2       ! Now determine and save the length of the data record.  Add it to the
2       ! overall length.
2
3       if not .sp[irc$v_rrv] then (
4               data_length =   (selectoneu .anl$gl_fat[fat$v_rtype] of set
4                               [fat$c_fixed]:          .anl$gl_fat[fat$w_maxrec];
4
4                               [fat$c_variable,
4                               fat$c_vfc]:             .rp[0,0,16,0];
3                               tes);
3               length = .length + .data_length;
2       );
2
2       ! Finally, make sure the entire thing fits.
2
3       if .b[bsd$l_offset]+.length gtru .hp[bkt$w_freespace] then (
3               anl$format_error(anlrms$_baddatarecfit,.b[bsd$l_vbn]);
3               signal (anlrms$_unwind);
2       );
```

```
2 ! Now we can actually format the structure, if requested.
2
3 if .report then (
3
3         ! We begin with a nice heading.
3
3         anl$format_line(3,.indent_level,anlrms$_idxprimrec,.b[bsd$l_vbn],.b[bsd$l_offset]);
3         anl$format_skip(0);
3
3         ! Now the control flags.
3
3         anl$format_flags(.indent_level+1,anlrms$_idxprimrecflags,.sp[irc$b_control],data_flags_def);
3
3         ! Now the record ID.
3
3         anl$format_line(0,.indent_level+1,anlrms$_idxprimrecid,.sp[irc$b_id]);
3
3         ! Now the RRV, both record ID and bucket pointer, if present.
3
3         if not .sp[irc$v_noptrsz] then
3                 anl$format_line(0,.indent_level+1,anlrms$_idxprimrecrrv,
3                                 .sp[irc$b_rrv_id],.sp[irc$v_ptrsz]+2,
4                                 (case .sp[irc$v_ptrsz] from 0 to 2 of set
4                                 [0]:    .sp[3,0,16,0];
4                                 [1]:    .sp[3,0,24,0];
4                                 [2]:    .sp[3,0,32,0];
3                                 tes));
3
3         ! Call a routine to format the primary key, if present.
3
4         if not .sp[irc$v_rrv] then (
4                 anl$format_line(0,.indent_level+1,anlrms$_idxkeybytes);
4                 anl$2format_primary_key(
4                         (if .anl$gl_fat[fat$v_rtype] nequ fat$c_fixed then .rp+2 else .rp),
4                         .key_bsd,.indent_level+2);
3         );
2 );
```

```
; 1074           1562  2 ! Now we can actually check the integrity of this data record.  Most of
; 1075           1563  2 ! the checking has been done, since it involved the fit of the record
; 1076           1564  2 ! in the bucket.  However, we have a few things to do.
; 1077           1565  2
; 1078           1566  2 ! Check the control bits, ignoring the pointer size.
; 1079           1567  2
; 1080           1568  2 anl$check_flags(.b[bsd$l_vbn],.sp[irc$b_control] and %x'fc',data_flags_def);
; 1081           1569  2
; 1082           1570  2 ! Now we can check the record length for VFC records to make sure they are
; 1083           1571  2 ! long enough to contain the header.
; 1084           1572  2
; 1085           1573  2 if not .sp[irc$v_rrv] then
; 1086           1574  2         if .anl$gl_fat[fat$v_rtype] eqlu fat$c_vfc and
; 1087           1575  2            .data_length lssu .anl$gl_fat[fat$b_vfcsize] then
; 1088           1576  2                 anl$format_error(anlrms$_vfctooshort,.b[bsd$l_vbn]);
; 1089           1577  2
; 1090         P 1578  2 if not .sp[irc$v_rrv] and not .sp[irc$v_deleted] then statistics_callback(
; 1091         P 1579  2
; 1092         P 1580  2         ! If we are accumulating statistics, we need to call the data
; 1093         P 1581  2         ! record callback routine, telling it the overall record length.
; 1094         P 1582  2
; 1095         P 1583  2         anl$data_callback(.data_length,
; 1096         P 1584  2                           0,
; 1097         P 1585  2                           0,
; 1098         P 1586  2                           0);
; 1099           1587  2 );
; 1100           1588  2
; 1101           1589  2 ! Now we want to advance on to the next data record.  If there is room in
; 1102           1590  2 ! the bucket for another, then update the BSD.  Otherwise don't touch it.
; 1103           1591  2
; 1104           1592  3 if .b[bsd$l_offset]+.length lssu .hp[bkt$w_freespace] then (
; 1105           1593  3         b[bsd$l_offset] = .b[bsd$l_offset] + .length;
; 1106           1594  3         return true;
; 1107           1595  2 ) else
; 1108           1596  2         return false;
; 1109           1597  2
; 1110           1598  1 end;
; INFO#212               L1:1492
; Null expression appears in value-required context


                                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

   44  45  54  45  4C  45  44  5F  56  24  43  52  49  0D  0018C P.ABC:   .ASCII  <13>\IRC$V_DELETED\          ;
                       56  52  52  5F  56  24  43  52  49  09  0019A P.ABD:   .ASCII  <9>\IRC$V_RRV\               ;
   5A  53  52  54  50  4F  4E  5F  56  24  43  52  49  0D  001A4 P.ABE:   .ASCII  <13>\IRC$V_NOPTRSZ\          ;

                                                                          .PSECT  $OWN$,NOEXE,2

                              00000000  00000000  00000004  000A0 DATA_FLAGS_DEF:
                                                                          .LONG   4, 0, 0                      ;
                              00000000' 00000000' 00000000' 000AC         .ADDRESS P.ABC, P.ABD, P.ABE         ;


                                                                          .PSECT  $CODE$,NOWRT,2
```

```
                                                    0FFC 00000       .ENTRY  ANL$2PRIMARY_DATA_RECORD, Save R2,R3,R4,R5,-: 1448
                                                                             R6,R7,R8,R9,R10,R11                     :
                                    5B  00000000G  00 9E 00002       MOVAB   LIB$SIGNAL, R11                         :
                                    5A  00000000G  8F D0 00009       MOVL    #ANLRMS$_UNWIND, R10                    :
                                    56         04  AC D0 00010       MOVL    REC_BSD, R6                             : 1451
                                    57         08  A6 7D 00014       MOVQ    8(R6), R7                               : 1475
           57     04    A8          10             00 ED 00018       CMPZV   #0, #16, 4(HP), R7                      :
                                                   13 1A 0001E       BGTRU   1$                                      :
                                               04  A6 DD 00020       PUSHL   4(R6)                                   : 1476
                                        00000000G  8F DD 00023       PUSHL   #ANLRMS$_BADDATARECFIT                  :
                            0000G   CF             02 FB 00029       CALLS   #2, ANL$FORMAT_ERROR                    :
                                                   5A DD 0002E       PUSHL   R10                                     : 1477
                                    6B             01 FB 00030       CALLS   #1, LIB$SIGNAL                          :
                        52          57         0C  A6 C1 00033 1$:   ADDL3   12(R6), R7, SP                          : 1483
                        33          62             04 E0 00038       BBS     #4, (SP), 7$                            : 1486
           53           62          02             00 EF 0003C       EXTZV   #0, #2, (SP), R3                        : 1487
                        03          00             53 CF 00041       CASEL   R3, #0, #3                              :
         0017         0012        000D           0008    00045 2$:   .WORD   3$-2$,-                                 :
                                                                             4$-2$,-                                 :
                                                                             5$-2$,-                                 :
                                                                             6$-2$                                   :
                                    55             03 D0 0004D 3$:   MOVL    #3, R5                                  :
                                                   1F 11 00050       BRB     8$                                      :
                                    55             04 D0 00052 4$:   MOVL    #4, R5                                  :
                                                   1A 11 00055       BRB     8$                                      :
                                    55             05 D0 00057 5$:   MOVL    #5, R5                                  :
                                                   15 11 0005A       BRB     8$                                      :
                                               04  A6 DD 0005C 6$:   PUSHL   4(R6)                                   : 1491
75                                      00000000G  8F DD 0005F       PUSHL   #ANLRMS$_BADDATARECPS                   :
                            0000G   CF             02 FB 00065       CALLS   #2, ANL$FORMAT_ERROR                    :
                                                   5A DD 0006A       PUSHL   R10                                     : 1492
78                                  6B             01 FB 0006C       CALLS   #1, LIB$SIGNAL                          :
95                                                 55 D4 0006F 7$:   CLRL    R5                                      : 1487
96                                  55             02 C0 00071 8$:   ADDL2   #2, LENGTH                              : 1485
                        54          52             55 C1 00074       ADDL3   LENGTH, SP, RP                          : 1495
00                      0C          62             03 E0 00078       BBS     #3, (SP), 9$                            : 1496
           01  0000G    DF          04             00 ED 0007C       CMPZV   #0, #4, @ANL$GL_FAT, #1                 :
                                                   03 13 00083       BEQL    9$                                      :
                                    55             02 C0 00085       ADDL2   #2, LENGTH                              : 1497
                        50          57             55 C1 00088 9$:   ADDL3   LENGTH, R7, R0                          : 1501
01         50     04    A8          10             00 ED 0008C       CMPZV   #0, #16, 4(HP), R0                      :
                                                   13 1E 00092       BGEQU   10$                                     :
05                                             04  A6 DD 00094       PUSHL   4(R6)                                   : 1502
                                        00000000G  8F DD 00097       PUSHL   #ANLRMS$_BADDATARECFIT                  :
                            0000G   CF             02 FB 0009D       CALLS   #2, ANL$FORMAT_ERROR                    :
                                                   5A DD 000A2       PUSHL   R10                                     : 1503
                                    6B             01 FB 000A4       CALLS   #1, LIB$SIGNAL                          :
                        2A          62             03 E0 000A7 10$:  BBS     #3, (SP), 15$                           : 1509
09                                  53      0000G  CF D0 000AB       MOVL    ANL$GL_FAT, R3                          : 1510
           50           63          04             00 EF 000B0       EXTZV   #0, #4, (R3), R0                        :
                                    01             50 D1 000B5       CMPL    R0, #1                                  : 1511
                                                   06 12 000B8       BNEQ    11$                                     :
 3                                  53         10  A3 3C 000BA       MOVZWL  16(R3), DATA_LENGTH                     :
                                                   12 11 000BE       BRB     14$                                     :
                                    02             50 D1 000C0 11$:  CMPL    R0, #2                                  : 1513
                                                   05 1F 000C3       BLSSU   12$                                     :
```

```
                                     03                 50  D1 000C5          CMPL     R0, #3                        ;
                                                        05  1B 000C8          BLEQU    13$                           ;
                                     53                 01  CE 000CA 12$:     MNEGL    #1, DATA_LENGTH               ;
                                                        03  11 000CD          BRB      14$                           ;
                                     53                 64  3C 000CF 13$:     MOVZWL   (RP), DATA_LENGTH             ; 1514
                                     55                 53  C0 000D2 14$:     ADDL2    DATA_LENGTH, LENGTH           ; 1516
                        59           57                 55  C1 000D5 15$:     ADDL3    LENGTH, R7, R9                ; 1521
          59      04    A8           10                 00  ED 000D9          CMPZV    #0, #16, 4(HP), R9            ;
                                                        13  1E 000DF          BGEQU    16$                           ;
                                                  04    A6  DD 000E1          PUSHL    4(R6)                         ; 1522
                                           00000000G    8F  DD 000E4          PUSHL    #ANLRMS$_BADDATARECFIT        ;
                                0000G    CF             02  FB 000EA          CALLS    #2, ANL$FORMAT_ERROR          ;
                                                        5A  DD 000EF          PUSHL    R10                           ; 1523
                                     6B                 01  FB 000F1          CALLS    #1, LIB$SIGNAL                ;
                                     03           0C    AC  E8 000F4 16$:     BLBS     REPORT, 17$                   ; 1527
                                                      00BA  31 000F8          BRW      26$                           ;
                                                        57  DD 000FB 17$:     PUSHL    R7                            ; 1531
                                                  04    A6  DD 000FD          PUSHL    4(R6)                         ;
                                           00000000G    8F  DD 00100          PUSHL    #ANLRMS$_IDXPRIMREC           ;
                                                  10    AC  DD 00106          PUSHL    INDENT_LEVEL                  ;
                                                        03  DD 00109          PUSHL    #3                            ;
                                0000G    CF             05  FB 0010B          CALLS    #5, ANL$FORMAT_LINE           ;
                                                        7E  D4 00110          CLRL     -(SP)                         ; 1532
                                0000G    CF             01  FB 00112          CALLS    #1, ANL$FORMAT_SKIP           ;
                                               0000'    CF  9F 00117          PUSHAB   DATA_FLAGS_DEF                ; 1536
                                     7E                 62  9A 0011B          MOVZBL   (SP), -(SP)                   ;
                                           00000000G    8F  DD 0011E          PUSHL    #ANLRMS$_IDXPRIMRECFLAGS      ;
                        57         10    AC             01  C1 00124          ADDL3    #1, INDENT_LEVEL, R7          ;
                                                        57  DD 00129          PUSHL    R7                            ;
                                0000G    CF             04  FB 0012B          CALLS    #4, ANL$FORMAT_FLAGS          ;
                                     7E           01    A2  9A 00130          MOVZBL   1(SP), -(SP)                  ; 1540
                                           00000000G    8F  DD 00134          PUSHL    #ANLRMS$_IDXPRIMRECID         ;
                                                        57  DD 0013A          PUSHL    R7                            ;
                                                        7E  D4 0013C          CLRL     -(SP)                         ;
                                0000G    CF             04  FB 0013E          CALLS    #4, ANL$FORMAT_LINE           ;
                        3B           62                 04  E0 00143          BBS      #4, (SP), 23$                 ; 1544
          50            62           02                 00  EF 00147          EXTZV    #0, #2, (SP), R0              ; 1547
                        02           00                 50  CF 0014C          CASEL    R0, #0, #2                    ;
                      0014         000C               0006     00150 18$:     .WORD    19$-18$,-                     ;
                                                                                       20$-18$,-                     ;
                                                                                       21$-18$                       ;
                                     7E           03    A2  3C 00156 19$:     MOVZWL   3(SP), -(SP)                  ; 1548
                                                        0B  11 0015A          BRB      22$                           ;
          7E      03    A2           18                 00  EF 0015C 20$:     EXTZV    #0, #24, 3(SP), -(SP)         ; 1549
                                                        03  11 00162          BRB      22$                           ;
                                                  03    A2  DD 00164 21$:     PUSHL    3(SP)                         ; 1550
          7E            62           02                 00  EF 00167 22$:     EXTZV    #0, #2, (SP), -(SP)           ; 1546
                                     6E                 02  C0 0016C          ADDL2    #2, (SP)                      ;
                                     7E           02    A2  9A 0016F          MOVZBL   2(SP), -(SP)                  ;
                                           00000000G    8F  DD 00173          PUSHL    #ANLRMS$_IDXPRIMRECRRV        ; 1545
                                                        57  DD 00179          PUSHL    R7                            ;
                                                        7E  D4 0017B          CLRL     -(SP)                         ;
                                0000G    CF             06  FB 0017D          CALLS    #6, ANL$FORMAT_LINE           ;
                        2F           62                 03  E0 00182 23$:     BBS      #3, (SP), 26$                 ; 1555
                                           00000000G    8F  DD 00186          PUSHL    #ANLRMS$_IDXKEYBYTES          ; 1556
                                                        57  DD 0018C          PUSHL    R7                            ;
                                                        7E  D4 0018E          CLRL     -(SP)                         ;
```

```
63                                          0000G  CF             03  FB 00190          CALLS   #3, ANL$FORMAT_LINE                           ;
                                     7E        10  AC             02  C1 00195          ADDL3   #2, INDENT_LEVEL, -(SP)                       ; 1559
                                                              08  AC  DD 0019A          PUSHL   KEY_BSD                                       ;
67                    01      0000G  DF            04             00  ED 0019D          CMPZV   #0, #4, @ANL$GL_FAT, #1                       ; 1558
                                                                  08  13 001A4          BEQL    24$                                           ;
                                                   50         02  A4  9E 001A6          MOVAB   2(R4), R0                                     ;
                                                                  50  DD 001AA          PUSHL   R0                                            ;
78                                                                02  11 001AC          BRB     25$                                           ;
                                                                  54  DD 001AE 24$:     PUSHL   RP                                            ;
                                            0000V  CF             03  FB 001B0 25$:     CALLS   #3, ANL$2FORMAT_PRIMARY_KEY                   ;
                                                           0000'  CF  9F 001B5 26$:     PUSHAB  DATA_FLAGS_DEF                                ; 1568
                                                   50             62  9A 001B9          MOVZBL  (SP), R0                                      ;
                                     7E            50   FFFFFF03  8F  CB 001BC          BICL3   #-253, R0, -(SP)                              ;
                                                              04  A6  DD 001C4          PUSHL   4(R6)                                         ;
                                            0000G  CF             03  FB 001C7          CALLS   #3, ANL$CHECK_FLAGS                           ;
                                     43            62             03  E0 001CC          BBS     #3, (SP), 29$                                 ; 1573
32                                                 50      0000G  CF  D0 001D0          MOVL    ANL$GL_FAT, R0                                ; 1574
33                    03             60            04             00  ED 001D5          CMPZV   #0, #4, (R0), #3                              ;
34                                                                16  12 001DA          BNEQ    27$                                           ;
                      53         0F  A0            08             00  ED 001DC          CMPZV   #0, #8, 15(R0), DATA_LENGTH                   ; 1575
                                                                  0E  1B 001E2          BLEQU   27$                                           ;
37                                                            04  A6  DD 001E4          PUSHL   4(R6)                                         ; 1576
                                                       00000000G  8F  DD 001E7          PUSHL   #ANLRMS$_VFCTOOSHORT                          ;
                                            0000G  CF             02  FB 001ED          CALLS   #2, ANL$FORMAT_ERROR                          ;
39                                   1D            62             03  E0 001F2 27$:     BBS     #3, (SP), 29$                                 ; 1578
                                     19            62             02  E0 001F6          BBS     #2, (SP), 29$                                 ;
                                                   02      0000G  CF  91 001FA          CMPB    ANL$GB_MODE, #2                               ; 1587
                                                                  07  13 001FF          BEQL    28$                                           ;
                                                   04      0000G  CF  91 00201          CMPB    ANL$GB_MODE, #4                               ;
                                                                  0B  12 00206          BNEQ    29$                                           ;
                                                                  7E  7C 00208 28$:     CLRQ    -(SP)                                         ;
                                                                  7E  D4 0020A          CLRL    -(SP)                                         ;
                                                                  53  DD 0020C          PUSHL   DATA_LENGTH                                   ;
                                            0000G  CF             04  FB 0020E          CALLS   #4, ANL$DATA_CALLBACK                         ;
                      59         04  A8            10             00  ED 00213 29$:     CMPZV   #0, #16, 4(HP), R9                            ; 1592
                                                                  08  1B 00219          BLEQU   30$                                           ;
                                               08  A6             55  C0 0021B          ADDL2   LENGTH, 8(R6)                                 ; 1593
                                                   50             01  D0 0021F          MOVL    #1, R0                                        ; 1596
                                                                      04 00222          RET                                                   ;
                                                                  50  D4 00223 30$:     CLRL    R0                                            ;
                                                                      04 00225          RET                                                   ; 1598

           ; Routine Size:  550 bytes,    Routine Base:  $CODE$ + 0927
```

```
: 1112          1599   1 %sbttl 'ANL$2FORMAT_PRIMARY_KEY - Format Primary Key from Data'
: 1113          1600   1 !++
: 1114          1601   1 ! Functional Description:
: 1115          1602   1 !       This routine is called to dump the primary key from a data
: 1116          1603   1 !       record in a prolog 2 indexed file.  This is more difficult than
: 1117          1604   1 !       prolog 3, because the primary key is not already extracted.
: 1118          1605   1 !
: 1119          1606   1 ! Formal Parameters:
: 1120          1607   1 !       rec_ptr         Pointer to data record.
: 1121          1608   1 !       key_bsd         Address of BSD describing key for this index.
: 1122          1609   1 !       indent_level    Indentation level for the report.
: 1123          1610   1 !
: 1124          1611   1 ! Implicit Inputs:
: 1125          1612   1 !       global data
: 1126          1613   1 !
: 1127          1614   1 ! Implicit Outputs:
: 1128          1615   1 !       global data
: 1129          1616   1 !
: 1130          1617   1 ! Returned Value:
: 1131          1618   1 !       none
: 1132          1619   1 !
: 1133          1620   1 ! Side Effects:
: 1134          1621   1 !
: 1135          1622   1 !--
: 1136          1623   1
: 1137          1624   1
: 1138          1625   2 global routine anl$2format_primary_key(rec_ptr,key_bsd,indent_level): novalue = begin
: 1139          1626   2
: 1140          1627   2 bind
: 1141          1628   2         k = .key_bsd: bsd;
: 1142          1629   2
: 1143          1630   2 local
: 1144          1631   2         kp: ref block[,byte],
: 1145          1632   2         segment: long,
: 1146          1633   2         buffer_i: long,
: 1147          1634   2         local_described_buffer(buffer,256);
: 1148          1635   2
: 1149          1636   2
: 1150          1637   2 ! Begin by setting up a pointer to the key descriptor.  Then define
: 1151          1638   2 ! a couple of arrays, one for the sizes and one for the positions.
: 1152          1639   2
: 1153          1640   2 kp = .k[bsd$l_bufptr] + .k[bsd$l_offset];
: 1154          1641   2
: 1155          1642   3 begin
: 1156          1643   3 bind
: 1157          1644   3         size_vector = kp[key$b_size0]: vector[,byte],
: 1158          1645   3         pos_vector = kp[key$w_position0]: vector[,word];
: 1159          1646   3
: 1160          1647   3 ' It's really pretty simple.  We loop through each of the key segments
: 1161          1648   3 ! and extract the data from the record.  The data is concatenated irto
: 1162          1649   3 ! the key buffer.
: 1163          1650   3
: 1164          1651   3 buffer[len] = 0;
: 1165          1652   3
: 1166          1653   4 incru segment from 0 to .kp[key$b_segments]-1 do (
: 1167          1654   4
: 1168          1655   4         ch$move(.size_vector[.segment],.rec_ptr+.pos_vector[.segment],
; 1169          1656  4                                        .buffer[ptr]+.buffer[len]);
; 1170          1657  4           buffer[len] = .buffer[len] + .size_vector[.segment];
; 1171          1658  3 );
; 1172          1659  2 end;
; 1173          1660  2
; 1174          1661  2 ! Now we can dump the key in hex.
; 1175          1662  2
; 1176          1663  2 anl$format_hex(.indent_level,buffer);
; 1177          1664  2
; 1178          1665  2 return;
; 1179          1666  2
; 1180          1667  1 end;



                                          01FC 00000         .ENTRY  ANL$2FORMAT_PRIMARY_KEY, Save R2,R3,R4,R5,- ; 1625
                                                                     R6,R7,R8                                    ;
                          5E      FEFC  CE  9E 00002         MOVAB   -260(SP), SP                                ;
                          50        08  AC  D0 00007         MOVL    KEY_BSD, R0                                 ; 1628
                          7E      0100  8F  3C 0000B         MOVZWL  #256, BUFFER                                ; 1634
                     04   AE        08  AE  9E 00010         MOVAB   BUFFER+8, BUFFER+4                          ;
             57      0C   A0        08  A0  C1 00015         ADDL3   8(R0), 12(R0), KP                           ; 1640
                                        6E  B4 0001B         CLRW    BUFFER                                      ; 1651
                          58        12  A7  9A 0001D         MOVZBL  18(KP), R8                                  ; 1653
                                        58  D7 00021         DECL    R8                                          ;
                                        56  D4 00023         CLRL    SEGMENT                                     ; 1655
                                        23  11 00025         BRB     2$                                          ;
                          52     2C A746  9A 00027 1$:       MOVZBL  44(KP)[SEGMENT], R2                         ;
                          51     1C A746  3C 0002C           MOVZWL  28(KP)[SEGMENT], R1                         ;
                          51        04  AC  C0 00031         ADDL2   REC_PTR, R1                                 ;
                          50            6E  3C 00035         MOVZWL  BUFFER, R0                                  ; 1656
                          50        04  AE  C0 00038         ADDL2   BUFFER+4, R0                                ;
             60           61            52  28 0003C         MOVC3   R2, (R1), (R0)                              ;
                          50     2C A746  9A 00040           MOVZBL  44(KP)[SEGMENT], R0                         ; 1657
                          6E            50  A0 00045         ADDW2   R0, BUFFER                                  ;
                                        56  D6 00048         INCL    SEGMENT                                     ; 1653
                          58            56  D1 0004A 2$:     CMPL    SEGMENT, R8                                 ;
                                        D8  1B 0004D         BLEQU   1$                                          ;
                                        5E  DD 0004F         PUSHL   SP                                          ; 1663
                                    0C  AC  DD 00051         PUSHL   INDENT_LEVEL                                ;
                  0000G   CF            02  FB 00054         CALLS   #2, ANL$FORMAT_HEX                          ;
                                            04 00059         RET                                                 ; 1667

; Routine Size:  90 bytes,    Routine Base:  $CODE$ + 0B4D
```

```
1 %sbttl 'ANL$2SIDR_RECORD - Print & Check A Secondary Data Record'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for printing and checking the contents
1 !       of a prolog 2 secondary data record.  Secondary data records exist
1 !       in the data buckets of secondary indices.  They co'tain SIDR records.
1 !
1 ! Formal Parameters:
1 !       rec_bsd         Address of BSD describing the data record.
1 !                       BSD is updated to point at next record.
1 !       key_bsd         Address of BSD describing the key for this index.
1 !       report          A boolean, true if we are to print the record.
1 !       indent_levei    Indentation level for the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       True if there is another SIDR in this bucket, false otherwise.
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$2sidr_record(rec_bsd,key_bsd,report,indent_level) = begin
2
2 bind
2       b = .rec_bsd: bsd,
2       k = .key_bsd: bsd;
2
2 own
2       sidr_flags_def: vector[6,long] initial(
2                               4,
2                               0,
2                               0,
2                               0,
2                               0,
2                               uplit byte (%ascic 'IRC$V_NODUPCNT')
2                               );
2
2 local
2       hp: ref block[,byte],
2       sp: ref block[,byte],
2       kp: ref block[,byte],
2       length: long,
2       p: bsd,
2       sidr_pointers;
2
2
2 ! First we have to ensure that the SIDR record fits in the used space of
2 ! the bucket.  If not, we have a drast'c structure error.  Begin by ensuring
2 ! that the first byte fits.
2
```

```
2 hp = .b[bsd$l_bufptr];
2
3 if .b[bsd$l_offset] gequ .hp[bkt$w_freespace] then (
3         anl$format_error(anlrms$_baddatarecfit,.b[bsd$l_vbn]);
3         signal (anlrms$_unwind);
2 );
2
2 ! Now we calculate the length of the entire SIDR record.
2
2 sp = .b[bsd$l_bufptr] + .b[bsd$l_offset];
2 length = 1 +
2         1 +
2         (if .sp[irc$v_nodupcnt] then 0 else 4) +
2         2 +
2         (if .sp[irc$v_nodupcnt] then .sp[2,0,16,0] else .sp[6,0,16,0]);
2
2 ! Make sure the record fits in the used portion of the bucket.
2
3 if .b[bsd$l_offset]+.length gtru .hp[bkt$w_freespace] then (
3         anl$format_error(anlrms$_baddatarecfit,.b[bsd$l_vbn]);
3         signal (anlrms$_unwind);
2 );
```

```
; 1262          1747  2 ! Now we can format the SIDR record fixed portion, if requested.
; 1263          1748  2
; 1264          1749  2 kp = .k[bsd$l_bufptr] + .k[bsd$l_offset];
; 1265          1750  3 if .report then (
; 1266          1751  3
; 1267          1752  3         . Start with a nice header.
. 1268          1753  3
; 1269          1754  3         anl$format_line(3,.indent_level,anlrms$_idxsidr,.b[bsd$l_vbn],.b[bsd$l_offset]);
; 1270          1755  3         anl$format_skip(0);
; 1271          1756  3
; 1272          1757  3         ! Now format the flags.
; 1273          1758  3
; 1274          1759  3         anl$format_flags(.indent_level+1,anlrms$_idxsidrflags,.sp[irc$b_control],sidr_flags_def);
; 1275          1760  3
; 1276          1761  3         ! Now format the record ID.
; 1277          1762  3
; 1278          1763  3         anl$format_line(0,.indent_level+1,anlrms$_idxsidrrecid,.sp[irc$b_id]);
; 1279          1764  3
; 1280          1765  3         ! Now format the duplicate count if it exists.
; 1281          1766  3
; 1282          1767  3         if not .sp[irc$v_nodupcnt] then
; 1283          1768  3                 anl$format_line(0,.indent_level+1,anlrms$_idxsidrdupcnt,.sp[2,0,32,0]);
; 1284          1769  3
; 1285          1770  3         ! Now the key.  We dump it in hex.
; 1286          1771  3
; 1287          1772  3         anl$format_line(0,.indent_level+1,anlrms$_idxkeybytes);
; 1288          1773  4         begin
; 1289          1774  4         local
; 1290          1775  4                 key_dsc: descriptor;
; 1291          1776  4
; 1292        P 1777  4         build_descriptor(key_dsc,.kp[key$b_keysz],
; 1293        P 1778  4                         .sp +
; 1294        P 1779  4                         1 +
; 1295        P 1780  4                         1 +
; 1296        P 1781  4                         (if .sp[irc$v_nodupcnt] then 0 else 4) +
; 1297          1782  4                         2);
; 1298          1783  4         anl$format_hex(.indent_level+2,key_dsc);
; 1299          1784  3         end;
; 1300          1785  2 );
```

```
  2 ! Now we can actually check the integrity of the SIDR record.  All we have
  2 ! to check is the flags.  Don't get confused by the pointer size bits.
  2
  2 anl$check_flags(.b[bsd$l_vbn],.sp[irc$b_control] and %x'fc',sidr_flags_def);
  2
  2 ! At this point, if we are formatting a report, we're done.  If we aren't
  2 ! (e.g., we are checking the file), then we want to check all of the
  2 ! SIDR pointers.
  2
  2 sidr_pointers = 0;
  3 if not .report then (
  3
  3         ! Set up a BSD to describe the first SIDR pointer.  This includes
  3         ! setting the work longword to the number of bytes worth of pointers
  3         ! existing in the record.
  3
  3         init_bsd(p);
  3         copy_bucket(b,p);
  3         p[bsd$l_offset] =        .b[bsd$l_offset] +
  3                                  1 +
  3                                  1 +
  3                                  (if .sp[irc$v_noptrsz] then 0 e.se 4) +
  3                                  2 +
  3                                  .kp[key$b_keysz];
  3         p[bsd$l_work] = (if .sp[irc$v_noptrsz] then .sp[2,0,16,0] else .sp[6,0,16,0]) -
  3                         .kp[key$b_keysz];
  3
  3         ! Now we can loop through each pointer, checking its integrity.
  3         ! We'll count them as we go.
  3
  3         do increment(sidr_pointers) while anl$2sidr_pointer(p,false);
  3
  3         anl$bucket(p,-1);
  2 );
  2
P 2 statistics_callback(
P 2
P 2         ! If we are accumulating statistics, we want to call the data
P 2         ! record callback routine and tell it the overall record length.
P 2         ! We also need to tell it the number of SIDR pointers in this record.
P 2
P 2         anl$data_callback(.length,
P 2                           0,
P 2                           0,
P 2                           .sidr_pointers);
  2 );
```

```
; 1349        1832  2 ! Now we want to advance on to the next SIDR in this bucket.  If there isn't
; 1350        1833  2 ! room for one, then we're done.  Otherwise update the BSD.
; 1351        1834  2
; 1352        1835  3 if .b[bsd$l_offset]+.length lssu .hp[bkt$w_freespace] then (
; 1353        1836  3         b[bsd$l_offset] = .b[bsd$l_offset] + .length;
; 1354        1837  3         return true;
; 1355        1838  2 ) else
; 1356        1839  2         return false;
; 1357        1840  2
; 1358        1841  1 end;



                                                                    .PSECT   $PLIT$,NOWRT,NOEXE,2

54  4E  43  50  55  44  4F  4E  5F  56  24  43  52  49  0E  001B2 P.ABF:  .ASCII   <14>\IRC$V_NODUPCNT\                        ;

                                                                    .PSECT   $OWN$,NOEXE,2

          00000000  00000000  00000000  00000000  00000004  000B8 SIDR_FLAGS_DEF:
                                                                    .LONG    4, 0, 0, 0, 0                               ;
                                                   00000000' 000CC  .ADDRESS P.ABF                                       ;



                                                                    .PSECT   $CODE$,NOWRT,2

                                                   0FFC 00000       .ENTRY   ANL$2SIDR_RECORD, Save R2,R3,R4,R5,R6,R7,-  ; 1696
                                                                             R8,R9,R10,R11                               ;
                                   5E             28 C2 00002       SUBL2    #40, SP                                     ;
                                   57        04   AC D0 00005       MOVL     REC_BSD, R7                                 ; 1699
                                   52        08   AC D0 00009       MOVL     KEY_BSD, R2                                 ; 1700
                                   59        0C   A7 D0 0000D       MOVL     12(R7), HP                                  ; 1725
                                   5A        08   A7 D0 00011       MOVL     8(R7), R10                                  ; 1727
          5A      04  A9           10             00 ED 00015       CMPZV    #0, #16, 4(HP), R10                         ;
                                                  1B 1A 0001B       BGTRU    1$                                          ;
                                             04   A7 DD 0001D       PUSHL    4(R7)                                       ; 1728
                                      00000000G   8F DD 00020       PUSHL    #ANLRMS$_BADDATARECFIT                      ;
                             0000G CF             02 FB 00026       CALLS    #2, ANL$FORMAT_ERROR                        ;
                                      00000000G   8F DD 0002B       PUSHL    #ANLRMS$_UNWIND                             ; 1729
                         00000000G 00             01 FB 00031       CALLS    #1, LIB$SIGNAL                              ;
                      56           5A        0C   A7 C1 00038 1$:   ADDL3    12(R7), R10, SP                             ; 1734
                      04           66             04 E1 0003D       BBC      #4, (SP), 2$                                ; 1737
                                                  50 D4 00041       CLRL     R0                                          ;
                                                  03 11 00043       BRB      3$                                          ;
                                   50             04 D0 00045 2$:   MOVL     #4, R0                                      ;
                      06           66             04 E1 00048 3$:   BBC      #4, (SP), 4$                                ; 1739
                                   51        02   A6 3C 0004C       MOVZWL   2(SP), R1                                   ;
                                                  04 11 00050       BRB      5$                                          ;
                                   51        06   A6 3C 00052 4$:   MOVZWL   6(SP), R1                                   ;
                                   6E        04 A140 9E 00056 5$:   MOVAB    4(R1)[R0], LENGTH                           ; 1738
                  04  AE           5A             6E C1 0005B       ADDL3    LENGTH, R10, 4(SP)                          ; 1743
      04  AE      04  A9           10             00 ED 00060       CMPZV    #0, #16, 4(HP), 4(SP)                       ;
                                                  1B 1E 00067       BGEQU    6$                                          ;
                                             04   A7 DD 00069       PUSHL    4(R7)                                       ; 1744
                                      00000000G   8F DD 0006C       PUSHL    #ANLRMS$_BADDATARECFIT                      ;
                             0000G CF             02 FB 00072       CALLS    #2, ANL$FORMAT_ERROR                        ;
```

```
                                   00000000G  8F DD 00077        PUSHL    #ANLRMS$_UNWIND                        ; 1745
             00000000G  00                    01 FB 0007D        CALLS    #1, LIB$SIGNAL                         ;
     58             0C  A2               08   A2 C1 00084 6$:    ADDL3    8(R2), 12(R2), KP                      ; 1749
                        03               0C   AC E8 0008A        BLBS     REPORT, 7$                             ; 1750
                                       0090   31 0008E           BRW      11$                                    ;
                                              5A DD 00091 7$:    PUSHL    R10                                    ; 1754
                                         04   A7 DD 00093        PUSHL    4(R7)                                  ;
                                  00000000G   8F DD 00096        PUSHL    #ANLRMS$_IDXSIDR                       ;
                                         10   AC DD 0009C        PUSHL    INDENT_LEVEL                           ;
                                              03 DD 0009F        PUSHL    #3                                     ;
                 0000G  CF                    05 FB 000A1        CALLS    #5, ANL$FORMAT_LINE                    ;
                                              7E D4 000A6        CLRL     -(SP)                                  ; 1755
                 0000G  CF                    01 FB 000A8        CALLS    #1, ANL$FORMAT_SKIP                    ;
                                      0000'   CF 9F 000AD        PUSHAB   SIDR_FLAGS_DEF                         ; 1759
                        7E                    66 9A 000B1        MOVZBL   (SP), -(SP)                            ;
                                  00000000G   8F DD 000B4        PUSHL    #ANLRMS$_IDXSIDRFLAGS                  ;
     52             10  AC                    01 C1 000BA        ADDL3    #1, INDENT_LEVEL, R2                   ;
                                              52 DD 000BF        PUSHL    R2                                     ;
                 0000G  CF                    04 FB 000C1        CALLS    #4, ANL$FORMAT_FLAGS                   ;
                        7E               01   A6 9A 000C6        MOVZBL   1(SP), -(SP)                           ; 1763
                                  00000000G   8F DD 000CA        PUSHL    #ANLRMS$_IDXSIDRRECID                  ;
                                              52 DD 000D0        PUSHL    R2                                     ;
                                              7E D4 000D2        CLRL     -(SP)                                  ;
                 0000G  CF                    04 FB 000D4        CALLS    #4, ANL$FORMAT_LINE                    ;
     12                 66                    04 E0 000D9        BBS      #4, (SP), 8$                           ; 1767
                                         02   A6 DD 000DD        PUSHL    2(SP)                                  ; 1768
                                  00000000G   8F DD 000E0        PUSHL    #ANLRMS$_IDXSIDRDUPCNT                 ;
                                              52 DD 000E6        PUSHL    R2                                     ;
                                              7E D4 000E8        CLRL     -(SP)                                  ;
                 0000G  CF                    04 FB 000EA        CALLS    #4, ANL$FORMAT_LINE                    ;
                                  00000000G   8F DD 000EF 8$:    PUSHL    #ANLRMS$_IDXKEYBYTES                   ; 1772
                                              52 DD 000F5        PUSHL    R2                                     ;
                                              7E D4 000F7        CLRL     -(SP)                                  ;
                 0000G  CF                    03 FB 000F9        CALLS    #3, ANL$FORMAT_LINE                    ;
                    08  AE               14   A8 9A 000FE        MOVZBL   20(KP), KEY_DSC                        ; 1782
     04                 66                    04 E1 00103        BBC      #4, (SP), 9$                           ;
                                              50 D4 00107        CLRL     R0                                     ;
                                              03 11 00109        BRB      10$                                    ;
                        50                    04 D0 0010B 9$:    MOVL     #4, R0                                 ;
                    0C  AE               04 A046 9E 0010E 10$:   MOVAB    4(R0)[SP], KEY_DSC+4                   ;
                                         08   AE 9F 00114        PUSHAB   KEY_DSC                                ; 1783
     7E             10  AC                    02 C1 00117        ADDL3    #2, INDENT_LEVEL, -(SP)                ;
                 0000G  CF                    02 FB 0011C        CALLS    #2, ANL$FORMAT_HEX                     ;
                                      0000'   CF 9F 00121 11$:   PUSHAB   SIDR_FLAGS_DEF                         ; 1789
                        50                    66 9A 00125        MOVZBL   (SP), R0                               ;
     7E                 50  FFFFFF03   8F CB 00128        BICL3    #-253, R0, -(SP)                       ;
                                         04   A7 DD 00130        PUSHL    4(R7)                                  ;
                 0000G  CF                    03 FB 00133        CALLS    #3, ANL$CHECK_FLAGS                    ;
                                              5B D4 0C138        CLRL     SIDR_POINTERS                          ; 1795
                        64               0C   AC E8 0013A        BLBS     REPORT, 17$                            ; 1796
18   00                 6E                    00 2C 0013E        MOVC5    #0, (SP), #0, #24, P                   ; 1802
                                         10   AE    00143                                                        ;
                    10  AE                    67 7D 00145        MOVQ     (R7), T                                ; 1803
                    18  AE               08   A7 D0 00149        MOVL     8(R7), T+8                             ;
                    24  AE               14   A7 D0 0014E        MOVL     20(R7), T+20                           ;
                                              7E D4 00153        CLRL     -(SP)                                  ;
                                         14   AE 9F 00155        PUSHAB   T                                      ;
```

```
                                0000G  CF            02  FB 00158         CALLS   #2, ANL$BUCKET                             ;
                         04            66            04  E1 0015D         BBC     #4, (SP), 12$                              ; 1807
                                                     50  D4 00161         CLRL    R0                                         ;
                                                     03  11 00163         BRB     13$                                        ;
                                       50            04  D0 00165 12$:    MOVL    #4, R0                                     ;
                                       50            5A  C0 00168 13$:    ADDL2   R10, R0                                    ; 1806
                                       51       14   A8  9A 0016B         MOVZBL  20(KP), R1                                 ; 1809
                                  18   AE       04 A140  9E 0016F         MOVAB   4(R1)[R0], P+8                             ; 1808
                         06            66            04  E1 00175         BBC     #4, (SP), 14$                              ; 1810
                                       56       02   A6  3C 00179         MOVZWL  2(SP), R6                                  ;
                                                     04  11 0017D         BRB     15$                                        ;
                                       56       06   A6  3C 0017F 14$:    MOVZWL  6(SP), R6                                  ;
                    24   AE            56            51  C3 00183 15$:    SUBL3   R1, R6, P+20                               ; 1811
                                                     5B  D6 00188 16$:    INCL    SIDR_POINTERS                              ; 1816
                                                     7E  D4 0018A         CLRL    -(SP)                                      ;
                                                14   AE  9F 0018C         PUSHAB  P                                          ;
                                0000V  CF            02  FB 0018F         CALLS   #2, ANL$2SIDR_POINTER                      ;
                                       F1            50  E8 00194         BLBS    R0, 16$                                    ;
                                       7E            01  CE 00197         MNEGL   #1, -(SP)                                  ; 1818
                                                14   AE  9F 0019A         PUSHAB  P                                          ;
                                0000G  CF            02  FB 0019D         CALLS   #2, ANL$BUCKET                             ;
                                       02     0000G  CF  91 001A2 17$:    CMPB    ANL$GB_MODE, #2                            ; 1831
                                                     07  13 001A7         BEQL    18$                                        ;
                                       04     0000G  CF  91 001A9         CMPB    ANL$GB_MODE, #4                            ;
                                                     0C  12 001AE         BNEQ    19$                                        ;
                                                     5B  DD 001B0 18$:    PUSHL   SIDR_POINTERS                              ;
                                                     7E  7C 001B2         CLRQ    -(SP)                                      ;
                                                0C   AE  DD 001B4         PUSHL   LENGTH                                     ;
                                0000G  CF            04  FB 001B7         CALLS   #4, ANL$DATA_CALLBACK                      ;
      04   AE       04   A9            10            00  ED 001BC 19$:    CMPZV   #0, #16, 4(HP), 4(SP)                      ; 1835
                                                     08  1B 001C3         BLEQU   20$                                        ;
                                  08   A7            6E  C0 001C5         ADDL2   LENGTH, 8(R7)                              ; 1836
                                       50            01  D0 001C9         MOVL    #1, R0                                     ; 1839
                                                         04 001CC         RET                                                ;
                                                     50  D4 001CD 20$:    CLRL    R0                                         ;
                                                         04 001CF         RET                                                ; 1841

; Routine Size:  464 bytes,    Routine Base:  $CODE$ + 0BA7
```

```
1 %sbttl 'ANL$2SIDR_POINTER - Format & Analyze SIDR Pointer'
1 !++
1 ! Functional Description:
1 !       This routine is responsible for formatting and analyzing one of the
1 !       pointers in a SIDR record for prolog 2 files.
1 !
1 ! Formal Parameters:
1 !       pointer_bsd     Address of BSD describing the pointer.  The work
1 !                       longword in the BSD is assumed to contain a count
1 !                       of remaining bytes in the SIDR record.
1 !       report          Boolean, true if we are to format the pointer.
1 !       indent_level    Indentation level for the report.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       True if there is another SIDR pointer, false otherwise.
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$2sidr_pointer(pointer_bsd,report,indent_level) = begin
2
2 bind
2         p = .pointer_bsd: bsd;
2
2 own
2         pointer_flags_def: vector[6,long] initial(
2                                 4,
2                                 0,
2                                 0,
2                                 uplit byte (%ascic 'IRC$V_DELETED'),
2                                 0,
2                                 uplit byte (%ascic 'IRC$V_NOPTRSZ')
2                                 );
2
2 local
2         pp: ref block[,byte],
2         length: long;
2
2
2 ! We know the SIDR record fits in the used space of the bucket, because
2 ! that was checked in ANL$2SIDR_RECORD.
2
2 ! So we can calculate the overall length of the pointer.
2
2 pp = .p[bsd$l_bufptr] + .p[bsd$l_offset];
2 length =        1 +
3                 (case .pp[irc$v_ptrsz] from 0 to 3 of set
3                 [0]:    3;
3                 [1]:    4;
; 1417          1899  3                 [2]:    5;
; 1418          1900  4                 [3]:    (anl$format_error(anlrms$_baddatarecps,.p[bsd$l_vbn]);
; 1419          1901  3                          signal (anlrms$_unwind););
; 1420          1902  2                 tes);
; 1421          1903  2
; 1422          1904  2 ! Make sure the entire pointer fits in the SIDR record.  If not, that's a
; 1423          1905  2 ! drastic structure error.
; 1424          1906  2
; 1425          1907  3 if .length gtru .p[bsd$l_work] then (
; 1426          1908  3         anl$format_error(anlrms$_badsidrptrfit,.p[bsd$l_vbn]);
; 1427          1909  3         signal (anlrms$_unwind);
; 1428          1910  2 );
; 1430        1911  2 ! Now we can format the SIDR pointer if requested.
; 1431        1912  2
; 1432        1913  3 if .report then (
; 1433        1914  3
; 1434        1915  3         ! Format the flags.
; 1435        1916  3
; 1436        1917  3         anl$format_flags(.indent_level,anlrms$_idxsidrptrflags,.pp[irc$b_control],pointer_flags_def);
; 1437        1918  3
; 1438        1919  3         ! And the record ID and bucket VBN.
; 1439        1920  3
; 1440        1921  3         anl$format_line(0,.indent_level,anlrms$_idxsidrptrref,.pp[1,0,8,0],.pp[irc$v_ptrsz]+2,
; 1441        1922  4                         (case .pp[irc$v_ptrsz] from 0 to 2 of set
; 1442        1923  4                         [0]:    .pp[2,0,16,0];
; 1443        1924  4                         [1]:    .pp[2,0,24,0];
; 1444        1925  4                         [2]:    .pp[2,0,32,0];
; 1445        1926  3                         tes));
; 1446        1927  2 );
; 1448          1928  2 ! Now we have to check the record pointer.  The only thing to check is
; 1449          1929  2 ! the control flags.  Don't get confused by the pointer size.
; 1450          1930  2
; 1451          1931  2 anl$check_flags(.p[bsd$l_vbn],.pp[irc$b_control] and %x'fc',pointer_flags_def);
; 1452          1932  2
; 1453          1933  2 ! Now we want to advance on to the next pointer.  Reduce the count of
; 1454          1934  2 ! remaining bytes.  If it goes to zero, there are no more pointers.
; 1455          1935  2 ! If it doesn't, then update the BSD.
; 1456          1936  2
; 1457          1937  2 p[bsd$l_work] = .p[bsd$l_work] - .length;
; 1458          1938  3 if .p[bsd$l_work] gtru 0 then (
; 1459          1939  3         p[bsd$l_offset] = .p[bsd$l_offset] + .length;
; 1460          1940  3         return true;
; 1461          1941  2 ) else
; 1462          1942  2         return false;
; 1463          1943  2
; 1464          1944  1 end;
; INFO#212                  L1:1901
; Null expression appears in value-required context



                                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

   44  45  54  45  4C  45  44  5F  56  24  43  52  49  0D  001C1 P.ABG:   .ASCII  <13>\IRC$V_DELETED\                    :
   5A  53  52  54  50  4F  4E  5F  56  24  43  52  49  0D  001CF P.ABH:   .ASCII  <13>\IRC$V_NOPTRSZ\                    :

                                                                                    .PSECT  $OWN$,NOEXE,2

                           00000000  00000000  00000004  000D0 POINTER_FLAGS_DEF:
                                                                                    .LONG   4, 0, 0                        :
                                               00000000' 000DC                      .ADDRESS P.ABG                         :
                                               00000000  000E0                      .LONG   0                              :
                                               00000000' 000E4                      .ADDRESS P.ABH                         :


                                                                                    .PSECT  $CODE$,NOWRT,2

                                                     00FC  00000                    .ENTRY  ANL$2SIDR_POINTER, Save R2,R3,R4,R5,R6,R7   : 1869
                                  57 00000000G  00   9E  00002                    MOVAB   LIB$SIGNAL, R7                      :
                                  56 00000000G  8F   D0  00009                    MOVL    #ANLRMS$_UNWIND, R6                 :
                                  54         04  AC   D0  00010                    MOVL    POINTER_BSD, R4                     : 1872
                    52       0C   A4         08  A4   C1  00014                    ADDL3   8(R4), 12(R4), PP                   : 1894
        55          62            02             00   EF  0001A                    EXTZV   #0, #2, (PP), R5                    : 1896
                    03            00             55   CF  0001F                    CASEL   R5, #0, #3                          :
      0017        0012          000D           0008       00023 1$:                .WORD   2$-1$,-                              :
                                                                                            3$-1$,-                              :
                                                                                            4$-1$,-                              :
                                                                                            5$-1$                                :
                                  53             03   D0  0002B 2$:                MOVL    #3, R3                              :
                                                 1F   11  0002E                    BRB     6$                                  :
                                  53             04   D0  00030 3$:                MOVL    #4, R3                              :
                                                 1A   11  00033                    BRB     6$                                  :
                                  53             05   D0  00035 4$:                MOVL    #5, R3                              :
                                                 15   11  00038                    BRB     6$                                  :
                                             04  A4   DD  0003A 5$:                PUSHL   4(R4)                               : 1900
```

```
                                          00000000G  8F  DD 0003D         PUSHL   #ANLRMS$_BADDATARECPS                       ;
                                0000G CF             02  FB 00043         CALLS   #2, ANL$FORMAT_ERROR                        ;
                                                     56  DD 00048         PUSHL   R6                                          ; 1901
                                      67             01  FB 0004A         CALLS   #1, LIB$SIGNAL                              ;
                                                     53  D4 0004D         CLRL    R3                                          ; 1896
                                                     53  D6 0004F 6$:     INCL    LENGTH                                      ; 1895
                                   14 A4             53  D1 00051         CMPL    LENGTH, 20(R4)                              ; 1907
                                                     13  1B 00055         BLEQU   7$                                          ;
                                                 04  A4  DD 00057         PUSHL   4(R4)                                       ; 1908
                                          00000000G  8F  DD 0005A         PUSHL   #ANLRMS$_BADSIDRPTRFIT                      ;
                                0000G CF             02  FB 00060         CALLS   #2, ANL$FORMAT_ERROR                        ;
                                                     56  DD 00065         PUSHL   R6                                          ; 1909
                                      67             01  FB 00067         CALLS   #1, LIB$SIGNAL                              ;
                                      51         08  AC  E9 0006A 7$:     BLBC    REPORT, 13$                                 ; 1913
                                              0000'  CF  9F 0006E         PUSHAB  POINTER_FLAGS_DEF                           ; 1917
                                      7E             62  9A 00072         MOVZBL  (PP), -(SP)                                 ;
                                          00000000G  8F  DD 00075         PUSHL   #ANLRMS$_IDXSIDRPTRFLAGS                    ;
                                                 0C  AC  DD 0007B         PUSHL   INDENT_LEVEL                                ;
                                0000G CF             04  FB 0007E         CALLS   #4, ANL$FORMAT_FLAGS                        ;
           50            62           02             00  EF 00083         EXTZV   #0, #2, (PP), R0                            ; 1922
                         02           00             50  CF 00088         CASEL   R0, #0, #2                                  ;
                       0014          000C          0006     0008C 8$:     .WORD   9$-8$,-                                     ;
                                                                                  10$-8$,-                                    ;
                                                                                  11$-8$                                      ;
                                      7E         02  A2  3C 00092 9$:     MOVZWL  2(PP), -(SP)                                ; 1923
                                                     0B  11 00096         BRB     12$                                         ;
           7E       02   A2                      18  00  EF 00098 10$:    EXTZV   #0, #24, 2(PP), -(SP)                       ; 1924
                                                     03  11 0009E         BRB     12$                                         ;
                                                 02  A2  DD 000A0 11$:    PUSHL   2(PP)                                       ; 1925
           7E            62           02             00  EF 000A3 12$:    EXTZV   #0, #2, (P, ), -(SP)                        ; 1921
                                      6E             02  C0 000A8         ADDL2   #2, (SP)                                    ;
                                      7E         01  A2  9A 000AB         MOVZBL  1(PP), -(SP)                                ;
                                          00000000G  8F  DD 000AF         PUSHL   #ANLRMS$_IDXSIDRPTRREF                      ;
                                                 0C  AC  DD 000B5         PUSHL   INDENT_LEVEL                                ;
                                                     7E  D4 000B8         CLRL    -(SP)                                       ;
                                0000G CF             06  FB 000BA         CALLS   #6, ANL$FORMAT_LINE                         ;
                                              0000'  CF  9F 000BF 13$:    PUSHAB  POINTER_FLAGS_DEF                           ; 1931
                                      50             62  9A 000C3         MOVZBL  (PP), R0                                    ;
                         7E           50  FFFFFF03   8F  CB 000C6         BICL3   #-253, R0, -(SP)                            ;
                                                 04  A4  DD 000CE         PUSHL   4(R4)                                       ;
                                0000G CF             03  FB 000D1         CALLS   #3, ANL$CHECK_FLAGS                         ;
                                   14 A4             53  C2 000D6         SUBL2   LENGTH, 20(R4)                              ; 1937
                                                     08  13 000DA         BEQL    14$                                         ; 1938
                                   08 A4             53  C0 000DC         ADDL2   LENGTH, 8(R4)                               ; 1939
                                      50             01  D0 000E0         MOVL    #1, R0                                      ; 1942
                                                         04 000E3         RET                                                 ;
                                                     50  D4 000E4 14$:    CLRL    R0                                          ;
                                                         04 000E6         RET                                                 ; 1944

; Routine Size:  231 bytes,    Routine Base:  $CODE$ + 0D77


; 1465          1945  1
; 1466          1946  0 end eludom
```

```
                                                                                              .EXTRN  LIB$SIGNAL

         ;                                        PSECT SUMMARY
         ;
         ;        Name                         Bytes                              Attributes
         ;
         ;   $CODE$                                3678  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
         ;   $PLIT$                                 477  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
         ;   $OWN$                                  232  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




         ;                                 Library Statistics
         ;
         ;                                                  -------- Symbols --------        Pages        Processing
5        ;        File                                      Total    Loaded   Percent        Mapped       Time
         ;
         ;   _$255$DUA28:[SYSLIB]LIB.L32;1                  18619       95          0        1000           00:01.8
8
4

0
1        ; Information:  3
3        ; Warnings:     0
         ; Errors:       0
5


         ;                                             COMMAND QUALIFIERS

6        ;        BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RMS2IDX/OBJ=OBJ$:RMS2IDX MSRC$:RMS2IDX/UPDATE=(ENH$:RMS2IDX)

         ; Size:          3678 code + 709 data bytes
7        ; Run Time:         01:01.6
         ; Elapsed Time:     03:11.5
3        ; Lines/CPU Min:      1896
         ; Lexemes/CPU-Min: 18683
         ; Memory Used:  399 pages
3        ; Compilation Complete

7
```

0007 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY