



```

RRRRRRRR      MM      MM      SSSSSSSS      SSSSSSSS      TTTTTTTTTT      AAAAAA      AAAAAA      TTTTTTTTTT      SSSSSSSS
RRRRRRRR      MM      MM      SSSSSSSS      SSSSSSSS      TTTTTTTTTT      AAAAAA      AAAAAA      TTTTTTTTTT      SSSSSSSS
RR      RR      MMMM      MMMM      SS      SS      TT      AA      AA      TT      SS
RR      RR      MMMM      MMMM      SS      SS      TT      AA      AA      TT      SS
RR      RR      MM      MM      SS      SS      TT      AA      AA      TT      SS
RRRRRRRR      MM      MM      SSSSSS      SSSSSS      TT      AA      AA      TT      SSSSSS
RRRRRRRR      MM      MM      SSSSSS      SSSSSS      TT      AA      AA      TT      SSSSSS
RR      RR      MM      MM      SS      SS      TT      AAAAAAAAAA      TT      SS
RR      RR      MM      MM      SS      SS      TT      AAAAAAAAAA      TT      SS
RR      RR      MM      MM      SS      SS      TT      AA      AA      TT      SS
RR      RR      MM      MM      SSSSSSSS      SSSSSSSS      TT      AA      AA      TT      SSSSSSSS
RR      RR      MM      MM      SSSSSSSS      SSSSSSSS      TT      AA      AA      TT      SSSSSSSS

```

```

....
....
....
....

```

```

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS

```

```
1 0001 0 %title 'RMSSTATS - Calculate and Report File Statistics'
2 0002 0 module rmsstats (
3 0003 1 ident='V04-000') = begin
4 0004 1
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
10 0010 1 * ALL RIGHTS RESERVED. *
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
17 0017 1 * TRANSFERRED. *
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
21 0021 1 * CORPORATION. *
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 Facility: VAX/VMS Analyze Facility, Calculate and Report File Statistics
32 0032 1
33 0033 1 Abstract: This module is responsible for accumulating, calculating, and
34 0034 1 reporting statistics about RMS files. This includes both the
35 0035 1 analysis primaries for FDL files, and statistics reports.
36 0036 1
37 0037 1
38 0038 1 Environment:
39 0039 1
40 0040 1 Author: Paul C. Anagnostopoulos, Creation Date: 11 August 1981
41 0041 1
42 0042 1 Modified By:
43 0043 1
44 0044 1 V03-005 PCA1012 Paul C. Anagnostopoulos 6-Apr-1983
45 0045 1 Do not generate data record compression statistic for
46 0046 1 secondary keys, because data record compression is
47 0047 1 meaningless for them.
48 0048 1 Duplicates per SIDR should count just duplicates, not
49 0049 1 including the first pointer.
50 0050 1 Add support for the index level 1 record count statistic.
51 0051 1
52 0052 1 V03-004 PCA1011 Paul C. Anagnostopoulos 1-Apr-1983
53 0053 1 Change the message prefix to ANLRMSS to ensure that
54 0054 1 message symbols are unique across all ANALYZEs. This
55 0055 1 is necessitated by the new merged message files.
56 0056 1
57 0057 1 V03-003 PCA0064 Paul Anagnostopoulos 29-Mar-1982
```

```
: 58      0058 1 | Use quadword arithmetic when calculating percentages  
: 59      0059 1 | so large files don't result in garbage numbers.  
: 60      0060 1 |  
: 61      0061 1 | V03-002 PCA0063      Paul Anagnostopoulos      29-Mar-1982  
: 62      0062 1 | Change calculation of index record statistics so they  
: 63      0063 1 | are parallel to data record statistics.  
: 64      0064 1 |  
: 65      0065 1 | V03-001 PCA0050      Paul Anagnostopoulos      26-Mar-1982  
: 66      0066 1 | Fix a bug in the way data record compression was  
: 67      0067 1 | calculated.  
: 68      0068 1 | --
```

```
: 70      0069 1 %sbttl 'Module Declarations'
: 71      0070 1
: 72      0071 1  | Libraries and Requires:
: 73      0072 1  |
: 74      0073 1  |
: 75      0074 1  | library 'lib';
: 76      0075 1  | require 'rmsreq';
: 77      0584 1
: 78      0585 1  |
: 79      0586 1  | Table of Contents:
: 80      0587 1  |
: 81      0588 1
: 82      0589 1 forward routine
: 83      0590 1     anl$fdl_analysis_of_area: novalue,
: 84      0591 1     anl$fdl_analysis_of_key: novalue,
: 85      0592 1     anl$area_statistics: novalue,
: 86      0593 1     anl$key_statistics: novalue,
: 87      0594 1     anl$percentage,
: 88      0595 1     anl$bucket_callback: novalue,
: 89      0596 1     anl$reclaimed_bucket_callback: novalue,
: 90      0597 1     anl$index_callback: novalue,
: 91      0598 1     anl$data_callback: novalue;
: 92      0599 1
: 93      0600 1  |
: 94      0601 1  | External References:
: 95      0602 1  |
: 96      0603 1
: 97      0604 1 external routine
: 98      0605 1     anl$format_line,
: 99      0606 1     anl$format_skip;
: 100     0607 1
: 101     0608 1 external
: 102     0609 1     anl$gl_fat: ref block[,byte],
: 103     0610 1     anl$gw_prolog: word;
```

```

: 105      0611  1  !
: 106      0612  1  ! Own Variables:
: 107      0613  1  !
: 108      0614  1  ! The following variables are used to accumulate statistical data about
: 109      0615  1  ! the file we are currently analyzing. The following abbreviations are
: 110      0616  1  ! used in the variable names:
: 111      0617  1  !
: 112      0618  1  !         blk      block
: 113      0619  1  !         c        count
: 114      0620  1  !         comp     compression
: 115      0621  1  !         len      length
: 116      0622  1  !         max      maximum
: 117      0623  1  !         rec      record
: 118      0624  1  !         t        total
: 119      0625  1  !
: 120      0626  1  ! own
: 121      0627  1  !         statistics_accumulators: vector[20,long] initial(rep 20 of (0));
: 122      0628  1  !
: 123      0629  1  ! ! This variable accumulates data about a single area of an indexed file.
: 124      0630  1  !
: 125      0631  1  ! bind
: 126      0632  1  !         reclaimed_blk_c           = statistics_accumulators[ 0]: long;
: 127      0633  1  !
: 128      0634  1  ! ! These variables accumulate data about a single key of an indexed file.
: 129      0635  1  ! ! There is a set of variables for the primary data level and a set for
: 130      0636  1  ! ! the index levels.
: 131      0637  1  !
: 132      0638  1  ! bind
: 133      0639  1  !         data_blk_c           = statistics_accumulators[ 1]: long,
: 134      0640  1  !         data_fill_t         = statistics_accumulators[ 2]: long,
: 135      0641  1  !         data_rec_c          = statistics_accumulators[ 3]: long,
: 136      0642  1  !         data_rec_len_t      = statistics_accumulators[ 4]: long,
: 137      0643  1  !         data_comp_key_len_t = statistics_accumulators[ 5]: long,
: 138      0644  1  !         data_comp_rec_len_t = statistics_accumulators[ 6]: long,
: 139      0645  1  !         data_sidr_ptr_c     = statistics_accumulators[ 7]: long,
: 140      0646  1  !
: 141      0647  1  !         index_blk_c         = statistics_accumulators[ 8]: long,
: 142      0648  1  !         index_fill_t        = statistics_accumulators[ 9]: long,
: 143      0649  1  !         index_rec_c         = statistics_accumulators[10]: long,
: 144      0650  1  !         index_level1_rec_c  = statistics_accumulators[11]: long,
: 145      0651  1  !         index_rec_len_t     = statistics_accumulators[12]: long,
: 146      0652  1  !         index_comp_rec_len_t = statistics_accumulators[13]: long,
: 147      0653  1  !         max_depth           = statistics_accumulators[14]: long;

```

```

: 149 0654 1 %sbttl 'ANL$FDL_ANALYSIS_OF_AREA - Generate ANALYSIS_OF_AREA Primary'
: 150 0655 1 ++
: 151 0656 1 Functional Description:
: 152 0657 1 This routine is responsible for generating an analysis primary for
: 153 0658 1 an indexed file area. This routine is called after the appropriate
: 154 0659 1 statistics have been accumulated via the statistics callback
: 155 0660 1 mechanism described below.
: 156 0661 1
: 157 0662 1 Formal Parameters:
: 158 0663 1 area_id The ID of this area.
: 159 0664 1
: 160 0665 1 Implicit Inputs:
: 161 0666 1 global data
: 162 0667 1
: 163 0668 1 Implicit Outputs:
: 164 0669 1 global data
: 165 0670 1
: 166 0671 1 Returned Value:
: 167 0672 1 none
: 168 0673 1
: 169 0674 1 Side Effects:
: 170 0675 1
: 171 0676 1 --
: 172 0677 1
: 173 0678 1
: 174 0679 2 global routine anl$fdl_analysis_of_area(area_id): novalue = begin
: 175 0680 2
: 176 0681 2
: 177 0682 2 ! We begin with a heading which includes the area ID.
: 178 0683 2
: 179 0684 2 anl$format_skip(0);
: 180 0685 2 anl$format_line(0,0,anlrms$_fdlanalarea,.area_id);
: 181 0686 2
: 182 0687 2 ! Now we put out the statistic that says how many blocks worth of
: 183 0688 2 ! reclaimed space there is.
: 184 0689 2
: 185 0690 2 anl$format_line(0,1,anlrms$_fdlrecl,.reclaimed_blk_c);
: 186 0691 2
: 187 0692 2 ! Clear the statistics accumulators for the next area.
: 188 0693 2
: 189 0694 2 ch$fill(%x'00', %allocation(statistics_accumulators),statistics_accumulators);
: 190 0695 2
: 191 0696 2 return;
: 192 0697 2
: 193 0698 1 end;

```

```

.TITLE RMSSTATS RMSSTATS - Calculate and Report File S
tatistics
.IDENT \V04-000\
.PSECT $OWNS,NOEXE,2

```

```

00000000# 0000 STATISTICS_ACCUMULATORS:
.LONG 0[20]

```

```

RECLAIMED_BLK_C= STATISTICS_ACCUMULATORS

```

```
DATA_BLK_C= STATISTICS_ACCUMULATORS+4
DATA_FILE_T= STATISTICS_ACCUMULATORS+8
DATA_REC_C= STATISTICS_ACCUMULATORS+12
DATA_REC_LEN_T= STATISTICS_ACCUMULATORS+16
DATA_COMP_KEY_LEN_T= STATISTICS_ACCUMULATORS+20
DATA_COMP_REC_LEN_T= STATISTICS_ACCUMULATORS+24
DATA_SDR_PTR_C= STATISTICS_ACCUMULATORS+28
INDEX_BLK_C= STATISTICS_ACCUMULATORS+32
INDEX_FILE_T= STATISTICS_ACCUMULATORS+36
INDEX_REC_C= STATISTICS_ACCUMULATORS+40
INDEX_LEVEL1_REC_C= STATISTICS_ACCUMULATORS+44
INDEX_REC_LEN_T= STATISTICS_ACCUMULATORS+48
INDEX_COMP_REC_LEN_T= STATISTICS_ACCUMULATORS+52
MAX_DEPTH= STATISTICS_ACCUMULATORS+56
.EXTRN ANLRM$$_OK, ANLRM$$_ALLOC
.EXTRN ANLRM$$_ANYTHING
.EXTRN ANLRM$$_BACKUP, ANLRM$$_BKT
.EXTRN ANLRM$$_BKTAREA
.EXTRN ANLRM$$_BKTCHK
.EXTRN ANLRM$$_BKTFLGS
.EXTRN ANLRM$$_BKTFREE
.EXTRN ANLRM$$_BKTKEY, ANLRM$$_BKTLEVEL
.EXTRN ANLRM$$_BKTNEXT
.EXTRN ANLRM$$_BKTPTRSIZE
.EXTRN ANLRM$$_BKTRECID
.EXTRN ANLRM$$_BKTRECID3
.EXTRN ANLRM$$_BKTSAMPLE
.EXTRN ANLRM$$_BKTVBNFREE
.EXTRN ANLRM$$_BUCKETSIZE
.EXTRN ANLRM$$_CELL, ANLRM$$_CELLDATA
.EXTRN ANLRM$$_CELLFLGS
.EXTRN ANLRM$$_CHKHGD
.EXTRN ANLRM$$_CONTIG, ANLRM$$_CREATION
.EXTRN ANLRM$$_CTLSIZE
.EXTRN ANLRM$$_DATAREC
.EXTRN ANLRM$$_DATABKTVBN
.EXTRN ANLRM$$_DUMPHEADING
.EXTRN ANLRM$$_EOF, ANLRM$$_ERRORCOUNT
.EXTRN ANLRM$$_ERRORNONE
.EXTRN ANLRM$$_ERRORS, ANLRM$$_EXPIRATION
.EXTRN ANLRM$$_FILEATR
.EXTRN ANLRM$$_FILEHDR
.EXTRN ANLRM$$_FILEID, ANLRM$$_FILEORG
.EXTRN ANLRM$$_FILESPEC
.EXTRN ANLRM$$_FLAG, ANLRM$$_GLOBALBUFS
.EXTRN ANLRM$$_HEXDATA
.EXTRN ANLRM$$_HEXHEADING1
.EXTRN ANLRM$$_HEXHEADING2
.EXTRN ANLRM$$_IDXAREA
.EXTRN ANLRM$$_IDXAREAALLOC
.EXTRN ANLRM$$_IDXAREABKTSZ
.EXTRN ANLRM$$_IDXAREANEXT
.EXTRN ANLRM$$_IDXAREANOALLOC
.EXTRN ANLRM$$_IDXAREAQTY
.EXTRN ANLRM$$_IDXAREARECL
.EXTRN ANLRM$$_IDXAREAUSED
```

.EXTRN ANLRM\$\$\_IDXKEY, ANLRM\$\$\_IDXKEYAREAS  
.EXTRN ANLRM\$\$\_IDXKEYBKTSZ  
.EXTRN ANLRM\$\$\_IDXKEYBYTES  
.EXTRN ANLRM\$\$\_IDXKEY1TYPE  
.EXTRN ANLRM\$\$\_IDXKEYDATAVBN  
.EXTRN ANLRM\$\$\_IDXKEYFILL  
.EXTRN ANLRM\$\$\_IDXKEYFLAGS  
.EXTRN ANLRM\$\$\_IDXKEYKEYSZ  
.EXTRN ANLRM\$\$\_IDXKEYNAME  
.EXTRN ANLRM\$\$\_IDXKEYNEXT  
.EXTRN ANLRM\$\$\_IDXKEYMINREC  
.EXTRN ANLRM\$\$\_IDXKEYNULL  
.EXTRN ANLRM\$\$\_IDXKEYPOSS  
.EXTRN ANLRM\$\$\_IDXKEYROOTLVL  
.EXTRN ANLRM\$\$\_IDXKEYROOTVBN  
.EXTRN ANLRM\$\$\_IDXKEYSEGS  
.EXTRN ANLRM\$\$\_IDXKEYSIZES  
.EXTRN ANLRM\$\$\_IDXPRIMREC  
.EXTRN ANLRM\$\$\_IDXPRIMRECFLAGS  
.EXTRN ANLRM\$\$\_IDXPRIMRECID  
.EXTRN ANLRM\$\$\_IDXPRIMRECLEN  
.EXTRN ANLRM\$\$\_IDXPRIMRECRV  
.EXTRN ANLRM\$\$\_IDXPROAREAS  
.EXTRN ANLRM\$\$\_IDXPROLOG  
.EXTRN ANLRM\$\$\_IDXREC, ANLRM\$\$\_IDXRECPT  
.EXTRN ANLRM\$\$\_IDXSIDR  
.EXTRN ANLRM\$\$\_IDXSIDRDUPCNT  
.EXTRN ANLRM\$\$\_IDXSIDRFLAGS  
.EXTRN ANLRM\$\$\_IDXSIDRRECID  
.EXTRN ANLRM\$\$\_IDXSIDRPTREFLAGS  
.EXTRN ANLRM\$\$\_IDXSIDRPTREF  
.EXTRN ANLRM\$\$\_INTERCOMMAND  
.EXTRN ANLRM\$\$\_INTERHDG  
.EXTRN ANLRM\$\$\_LONGREC  
.EXTRN ANLRM\$\$\_MAXRECSIZE  
.EXTRN ANLRM\$\$\_NOBACKUP  
.EXTRN ANLRM\$\$\_NOEXPIRATION  
.EXTRN ANLRM\$\$\_NOSPANFILLER  
.EXTRN ANLRM\$\$\_PERFORM  
.EXTRN ANLRM\$\$\_PROLOGFLAGS  
.EXTRN ANLRM\$\$\_PROLOGVER  
.EXTRN ANLRM\$\$\_PROT, ANLRM\$\$\_RECATTR  
.EXTRN ANLRM\$\$\_RECFMT, ANLRM\$\$\_RECLAIMBKT  
.EXTRN ANLRM\$\$\_RELBUCKET  
.EXTRN ANLRM\$\$\_RELEOFVBN  
.EXTRN ANLRM\$\$\_RELMAXREC  
.EXTRN ANLRM\$\$\_RELPROLOG  
.EXTRN ANLRM\$\$\_RELIAB, ANLRM\$\$\_REVISION  
.EXTRN ANLRM\$\$\_STATHDG  
.EXTRN ANLRM\$\$\_SUMMARYHDG  
.EXTRN ANLRM\$\$\_OWNERUIC  
.EXTRN ANLRM\$\$\_JNL, ANLRM\$\$\_AIJNL  
.EXTRN ANLRM\$\$\_BIJNL, ANLRM\$\$\_ATJNL  
.EXTRN ANLRM\$\$\_ATTOP, ANLRM\$\$\_BADCMD  
.EXTRN ANLRM\$\$\_BADPATH  
.EXTRN ANLRM\$\$\_BADVBN, ANLRM\$\$\_DOWNHELP  
.EXTRN ANLRM\$\$\_DOWNPATH

.EXTRN ANLRMSS\_EMPTYBKT  
.EXTRN ANLRMSS\_NODATA, ANLRMSS\_NODOWN  
.EXTRN ANLRMSS\_NONEXT, ANLRMSS\_NORECLAIMED  
.EXTRN ANLRMSS\_NORECS, ANLRMSS\_NORRV  
.EXTRN ANLRMSS\_RESTDONE  
.EXTRN ANLRMSS\_STACKFULL  
.EXTRN ANLRMSS\_UNINITINDEX  
.EXTRN ANLRMSS\_FDLIDENT  
.EXTRN ANLRMSS\_FDLSYSTEM  
.EXTRN ANLRMSS\_FDLSOURCE  
.EXTRN ANLRMSS\_FDLFILE  
.EXTRN ANLRMSS\_FDLALLOC  
.EXTRN ANLRMSS\_FDLNOALLOC  
.EXTRN ANLRMSS\_FDLBESTTRY  
.EXTRN ANLRMSS\_FDLBUCKETSIZE  
.EXTRN ANLRMSS\_FDLCLUSTERSIZE  
.EXTRN ANLRMSS\_FDLCONTIG  
.EXTRN ANLRMSS\_FDLXTENSION  
.EXTRN ANLRMSS\_FDLGLOBALBUFS  
.EXTRN ANLRMSS\_FDLMAXRECORD  
.EXTRN ANLRMSS\_FDLFILENAME  
.EXTRN ANLRMSS\_FDLORG, ANLRMSS\_FDLOWNER  
.EXTRN ANLRMSS\_FDLPROTECTION  
.EXTRN ANLRMSS\_FDLRECORD  
.EXTRN ANLRMSS\_FDLSPAN  
.EXTRN ANLRMSS\_FDLCC, ANLRMSS\_FDLVFCSIZE  
.EXTRN ANLRMSS\_FDLFORMAT  
.EXTRN ANLRMSS\_FDLsize  
.EXTRN ANLRMSS\_FDLAREA  
.EXTRN ANLRMSS\_FDLKEY, ANLRMSS\_FDLCHANGES  
.EXTRN ANLRMSS\_FDLDATAAREA  
.EXTRN ANLRMSS\_FDLDATAFILL  
.EXTRN ANLRMSS\_FDLDATAKEYCOMP  
.EXTRN ANLRMSS\_FDLDATARECCOMP  
.EXTRN ANLRMSS\_FDLDUPS  
.EXTRN ANLRMSS\_FDLINDEXAREA  
.EXTRN ANLRMSS\_FDLINDEXCOMP  
.EXTRN ANLRMSS\_FDLINDEXFILL  
.EXTRN ANLRMSS\_FDLINDEXAREA  
.EXTRN ANLRMSS\_FDLKEYNAME  
.EXTRN ANLRMSS\_FDLNORECS  
.EXTRN ANLRMSS\_FDLNULLKEY  
.EXTRN ANLRMSS\_FDLNULLVALUE  
.EXTRN ANLRMSS\_FDLPROLOG  
.EXTRN ANLRMSS\_FDLSEGLENGTH  
.EXTRN ANLRMSS\_FDLSEGPOS  
.EXTRN ANLRMSS\_FDLSEGTYPE  
.EXTRN ANLRMSS\_FDLANALAREA  
.EXTRN ANLRMSS\_FDLRECL  
.EXTRN ANLRMSS\_FDLANALKEY  
.EXTRN ANLRMSS\_FDLDATAKEYCOMP  
.EXTRN ANLRMSS\_FDLDATARECCOMP  
.EXTRN ANLRMSS\_FDLDATARECS  
.EXTRN ANLRMSS\_FDLDATASPACE  
.EXTRN ANLRMSS\_FDLDEPTH  
.EXTRN ANLRMSS\_FDLDUPSPER  
.EXTRN ANLRMSS\_FDLIDXCOMP

.EXTRN ANLRMSS\_FDLIDXFILL  
.EXTRN ANLRMSS\_FDLIDXSPACE  
.EXTRN ANLRMSS\_FDLIDL1RECS  
.EXTRN ANLRMSS\_FDLDATALENMEAN  
.EXTRN ANLRMSS\_FDLIDXLENMEAN  
.EXTRN ANLRMSS\_STATAREA  
.EXTRN ANLRMSS\_STATRECL  
.EXTRN ANLRMSS\_STATKEY  
.EXTRN ANLRMSS\_STATDEPTH  
.EXTRN ANLRMSS\_STATIDL1RECS  
.EXTRN ANLRMSS\_STATIDXLENMEAN  
.EXTRN ANLRMSS\_STATIDXSPACE  
.EXTRN ANLRMSS\_STATIDXFILL  
.EXTRN ANLRMSS\_STATIDXCOMP  
.EXTRN ANLRMSS\_STATDATARECS  
.EXTRN ANLRMSS\_STATDUPSPER  
.EXTRN ANLRMSS\_STATDATALENMEAN  
.EXTRN ANLRMSS\_STATDATASPACE  
.EXTRN ANLRMSS\_STATDATAFILL  
.EXTRN ANLRMSS\_STATDATAKEYCOMP  
.EXTRN ANLRMSS\_STATDATARECCOMP  
.EXTRN ANLRMSS\_STATEFFICIENCY  
.EXTRN ANLRMSS\_BADAREA1ST2  
.EXTRN ANLRMSS\_BADAREABKTSIZE  
.EXTRN ANLRMSS\_BADAREAFIT  
.EXTRN ANLRMSS\_BADAREAID  
.EXTRN ANLRMSS\_BADAREANEXT  
.EXTRN ANLRMSS\_BADAREAROOT  
.EXTRN ANLRMSS\_BADAREAUSED  
.EXTRN ANLRMSS\_BADBKTAREAID  
.EXTRN ANLRMSS\_BADBKTCHECK  
.EXTRN ANLRMSS\_BADBKTFREE  
.EXTRN ANLRMSS\_BADBKTKEYID  
.EXTRN ANLRMSS\_BADBKTLEVEL  
.EXTRN ANLRMSS\_BADBKTROOTBIT  
.EXTRN ANLRMSS\_BADBKTSAMPLE  
.EXTRN ANLRMSS\_BADCELLFIT  
.EXTRN ANLRMSS\_BADCHECKSUM  
.EXTRN ANLRMSS\_BADDATARECBITS  
.EXTRN ANLRMSS\_BADDATARECFIT  
.EXTRN ANLRMSS\_BADDATARECPS  
.EXTRN ANLRMSS\_BAD3IDXKEYFIT  
.EXTRN ANLRMSS\_BADIDLXLASTKEY  
.EXTRN ANLRMSS\_BADIDXORDER  
.EXTRN ANLRMSS\_BADIDXRECBITS  
.EXTRN ANLRMSS\_BADIDXRECFIT  
.EXTRN ANLRMSS\_BADIDXRECPS  
.EXTRN ANLRMSS\_BADKEYAREAID  
.EXTRN ANLRMSS\_BADKEYDATABKT  
.EXTRN ANLRMSS\_BADKEYDATAFIT  
.EXTRN ANLRMSS\_BADKEYDATATYPE  
.EXTRN ANLRMSS\_BADKEYIDXBKT  
.EXTRN ANLRMSS\_BADKEYFILL  
.EXTRN ANLRMSS\_BADKEYFIT  
.EXTRN ANLRMSS\_BADKEYREFID  
.EXTRN ANLRMSS\_BADKEYROOTLEVEL  
.EXTRN ANLRMSS\_BADKEYSEGCOUNT

```
.EXTRN ANLRMSS_BADKEYSEGVEC
.EXTRN ANLRMSS_BADKEYSUMMARY
.EXTRN ANLRMSS_BADREADNOPAR
.EXTRN ANLRMSS_BADREADPAR
.EXTRN ANLRMSS_BADSIDRDUPCT
.EXTRN ANLRMSS_BADSIDRPTRFIT
.EXTRN ANLRMSS_BADSIDRPTRSZ
.EXTRN ANLRMSS_BADSIDRSIZE
.EXTRN ANLRMSS_BADSTREAMEOF
.EXTRN ANLRMSS_BADVBNFREE
.EXTRN ANLRMSS_BKTLOOP
.EXTRN ANLRMSS_EXTENDERR
.EXTRN ANLRMSS_FLAGERROR
.EXTRN ANLRMSS_MISSINGBKT
.EXTRN ANLRMSS_NOTOK, ANLRMSS_SPANERROR
.EXTRN ANLRMSS_TOOMANYRECS
.EXTRN ANLRMSS_UNWIND, ANLRMSS_VFCTOOSHORT
.EXTRN ANLRMSS_CACHEFULL
.EXTRN ANLRMSS_CACHERELFAIL
.EXTRN ANLRMSS_FACILITY
.EXTRN ANL$FORMAT_LINE
.EXTRN ANL$FORMAT_SKIP
.EXTRN ANL$GL_FAT, ANL$GW_PROLOG
```

.PSECT \$CODE\$,NOWRT,2

```
.ENTRY ANL$FDL_ANALYSIS_OF_AREA, Save R2,R3,R4,R5 : 0679
CLRL -(SP) : 0684
CALLS #1, ANL$FORMAT_SKIP :
PUSHL AREA_ID : 0685
PUSHL #ANLRMSS_FDLANALAREA :
CLRQ -(SP) :
CALLS #4, ANL$FORMAT_LINE :
PUSHL RECLAIMED_BLK_C : 0690
PUSHL #ANLRMSS_FDLRECL :
PUSHL #1 :
CLRL -(SP) :
CALLS #4, ANL$FORMAT_LINE :
MOVCS #0, (SP), #0, #80, STATISTICS_ACCUMULATORS : 0694
RET : 0698
```

```
003C 00000
0000G CF 7E D4 00002
04 AC DD 00004
00000000G 8F DD 00009
0000G CF 7E 7C 00012
0000' 04 FB 00014
00000000G CF DD 00019
0000' 8F DD 0001D
0000G CF 01 DD 00023
0000' 7E D4 00025
0050 8F 00 04 FB 00027
6E 00 2C 0002C
0000' CF 00033
04 00036
```

; Routine Size: 55 bytes, Routine Base: \$CODE\$ + 0000

```
195 0699 1 %sbttl 'ANL$FDL_ANALYSIS_OF_KEY - Generate FDL Analysis Primary'
196 0700 1 ++
197 0701 1 Functional Description:
198 0702 1 This routine is responsible for generating an analysis primary for
199 0703 1 an indexed file key. This routine is called after the appropriate
200 0704 1 statistics have been accumulated via the statistics callback
201 0705 1 mechanism described below.
202 0706 1
203 0707 1 Formal Parameters:
204 0708 1 key_bsd Address of BSD describing the key.
205 0709 1
206 0710 1 Implicit Inputs:
207 0711 1 global data
208 0712 1
209 0713 1 Implicit Outputs:
210 0714 1 global data
211 0715 1
212 0716 1 Returned Value:
213 0717 1 none
214 0718 1
215 0719 1 Side Effects:
216 0720 1
217 0721 1 --
218 0722 1
219 0723 1
220 0724 2 global routine anl$fdl_analysis_of_key(key_bsd): novalue = begin
221 0725 2
222 0726 2 bind
223 0727 2 k = .key_bsd: bsd;
224 0728 2
225 0729 2 local
226 0730 2 kp: ref block[,byte],
227 0731 2 data_bytes: long;
228 0732 2
229 0733 2
230 0734 2 ! We begin with a heading which includes the key ID.
231 0735 2
232 0736 2 kp = .k[bsd$l_bufptr] + .k[bsd$l_offset];
233 0737 2 anl$format_skip(0);
234 0738 2 anl$format_line(0,0,anlrms$_fdlanalkey,.kp[key$b_keyref]);
235 0739 2
236 0740 2 ! If there are no data records in this index because it is uninitialized,
237 0741 2 ! then let's just put out a comment and quit. Otherwise we'll have a
238 0742 2 ! lot of trouble with dividing by zero.
239 0743 2
240 0744 3 if .data_rec_c eq 0 then (
241 0745 3 anl$format_line(0,1,anlrms$_fdlnorecs);
242 0746 3 return;
243 0747 2 );
244 0748 2
245 0749 2 ! Now we put out the statistics for this index in alphabetical order.
246 0750 2 ! Some items only apply to primary or secondary indices. Others only
247 0751 2 ! apply to prologue 3.
248 0752 2
249 0753 2 anl$format_line(0,1,anlrms$_fdldatafill,
250 0754 2 anl$percentage(.data_fill_t,.data_blk_c*512));
251 0755 3 if .anl$gw_prolog eq 0 then (
```

```

: 252 0756 3      anl$format_line(0,1,anlrms$_fdldatakeycomp,
: 253 0757 3      anl$percentage(.data_rec_c*.kp[key$b_keysz] - .data_comp_key_len_t,.data_rec_c*.kp[key$b_key
: 254 0758 4      if .kp[key$b_keyref] eqlu 0 then (
: 255 0759 4      data_bytes = .data_rec_len_t - .data_rec_c*.kp[key$b_keysz];
: 256 0760 4      anl$format_line(0,T,anlrms$_fdldatareccomp,
: 257 0761 4      anl$percentage(.data_bytes - .data_comp_rec_len_t,.data_bytes));
: 258 0762 3      );
: 259 0763 2  );
: 260 0764 2  anl$format_line(0,1,anlrms$_fdldatarecs,
: 261 0765 2  .data_rec_c);
: 262 0766 2  anl$format_line(0,1,anlrms$_fdldataspace,
: 263 0767 2  .data_blk_c);
: 264 0768 2  anl$format_line(0,1,anlrms$_fdldepth,
: 265 0769 2  .max_depth);
: 266 0770 2  if .kp[key$b_keyref] nequ 0 then
: 267 0771 2  anl$format_line(0,1,anlrms$_fdldupsper,
: 268 0772 2  (.data_sldr_ptr_c + .data_rec_c/2) /
: 269 0773 2  .data_rec_c - 1);
: 270 0774 2  if .anl$gw_prolog eqlu plg$cvr_3 then
: 271 0775 2  anl$format_line(0,1,anlrms$_fdlidxcomp,
: 272 0776 2  anl$percentage(.index_rec_len_t - .index_comp_rec_len_t,.index_rec_len_t));
: 273 0777 2  anl$format_line(0,1,anlrms$_fdlidxfill,
: 274 0778 2  anl$percentage(.index_fill_t,.index_blk_c*512));
: 275 0779 2  anl$format_line(0,1,anlrms$_fdlidxspace,
: 276 0780 2  .index_blk_c);
: 277 0781 2  anl$format_line(0,T,anlrms$_fdlidxl1recs,
: 278 0782 2  .index_level1_rec_c);
: 279 0783 2  anl$format_line(0,1,anlrms$_fdldataalenmean,
: 280 0784 2  (.data_rec_len_t + .data_rec_c/2) /
: 281 0785 2  .data_rec_c);
: 282 0786 2  anl$format_line(0,1,anlrms$_fdlidxlenmean,
: 283 0787 2  (.index_rec_len_t + .index_rec_c/2) /
: 284 0788 2  .index_rec_c);
: 285 0789 2
: 286 0790 2  ! Clear the statistics accumulators for the next key.
: 287 0791 2
: 288 0792 2  ch$fill(%x'00', %allocation(statistics_accumulators),statistics_accumulators);
: 289 0793 2
: 290 0794 2  return;
: 291 0795 2
: 292 0796 1  end;

```

				01FC 00000	.ENTRY	ANL\$FDL_ANALYSIS_OF_KEY, Save R2,R3,R4,R5,- ;	0724
						R6,R7,R8	
		58	0000V	CF 9E 00002	MOVAB	ANL\$PERCENTAGE, R8	
		57	0000G	CF 9E 00007	MOVAB	ANL\$FORMAT_LINE, R7	
		56	0000'	CF 9E 0000C	MOVAB	DATA_REC_C, R6	
		50	04	AC D0 00011	MOVL	KEY_BSD, R0	0727
52	0C	A0	08	A0 C1 00015	ADDL3	8(R0), 12(R0), KP	0736
				7E D4 0001B	CLRL	-(SP)	0737
	0000G	CF		01 FB 0001D	CALLS	#1, ANL\$FORMAT_SKIP	
		7E	15	A2 9A 00022	MOVZBL	21(KP), -(SP)	0738
			00000000G	8F DD 00026	PUSHL	#ANLRMS\$_FDLANALKEY	

			7E	7C	0002C	CLRQ	-(SP)		
	67		04	FB	0002E	CALLS	#4, ANL\$FORMAT_LINE		0744
			66	D5	00031	TSTL	DATA_REC_C		
		00000000G	0E	12	00033	BNEQ	1\$		0745
			8F	DD	00035	PUSHL	#ANLRMSS_FDLNORECS		
			01	DD	0003B	PUSHL	#1		
	67		7E	D4	0003D	CLRL	-(SP)		
			03	FB	0003F	CALLS	#3, ANL\$FORMAT_LINE		0744
			04		00042	RET			0754
7E	F8	A6	09	78	00043	1\$: ASHL	#9, DATA_BLK_C, -(SP)		
		FC	A6	DD	00048	PUSHL	DATA_FILE_T		
	68		02	FB	0004B	CALLS	#2, ANL\$PERCENTAGE		
		00000000G	50	DD	0004E	PUSHL	R0		0753
			8F	DD	00050	PUSHL	#ANLRMSS_FDLDATAFILL		
			01	DD	00056	PUSHL	#1		
	67		7E	D4	00058	CLRL	-(SP)		
	03	0000G	04	FB	0005A	CALLS	#4, ANL\$FORMAT_LINE		0755
			CF	B1	0005D	CMPL	ANL\$GW_PROLOG, #3		
	50	14	4A	12	00062	BNEQ	2\$		0757
	50		A2	9A	00064	MOVZBL	20(KP), R0		
			66	C4	00068	MULL2	DATA_REC_C, R0		
			50	DD	0006B	PUSHL	R0		
7E	50	08	A6	C3	0006D	SUBL3	DATA_COMP_KEY_LEN_T, R0, -(SP)		
	68		02	FB	00072	CALLS	#2, ANL\$PERCENTAGE		
		00000000G	50	DD	00075	PUSHL	R0		0756
			8F	DD	00077	PUSHL	#ANLRMSS_FDLDATAKEYCOMP		
			01	DD	0007D	PUSHL	#1		
			7E	D4	0007F	CLRL	-(SP)		
	67		04	FB	00081	CALLS	#4, ANL\$FORMAT_LINE		0758
		15	A2	95	00084	TSTB	21(KP)		
			25	12	00087	BNEQ	2\$		
	50	14	A2	9A	00089	MOVZBL	20(KP), R0		0759
	50		66	C4	0008D	MULL2	DATA_REC_C, R0		
50	04	A6	50	C3	00090	SUBL3	R0, DATA_REC_LEN_T, DATA_BYTES		
			50	DD	00095	PUSHL	DATA_BYTES		0761
7E	50	0C	A6	C3	00097	SUBL3	DATA_COMP_REC_LEN_T, DATA_BYTES, -(SP)		
	68		02	FB	0009C	CALLS	#2, ANL\$PERCENTAGE		
		00000000G	50	DD	0009F	PUSHL	R0		0760
			8F	DD	000A1	PUSHL	#ANLRMSS_FDLDATARECCOMP		
			01	DD	000A7	PUSHL	#1		
			7E	D4	000A9	CLRL	-(SP)		
	67		04	FB	000AB	CALLS	#4, ANL\$FORMAT_LINE		0765
		00000000G	66	DD	000AE	2\$: PUSHL	DATA_REC_C		0764
			8F	DD	000B0	PUSHL	#ANLRMSS_FDLDATARECS		
			01	DD	000B6	PUSHL	#1		
			7E	D4	000B8	CLRL	-(SP)		
	67		04	FB	000BA	CALLS	#4, ANL\$FORMAT_LINE		0767
		F8	A6	DD	000BD	PUSHL	DATA_BLK_C		
		00000000G	8F	DD	000C0	PUSHL	#ANLRMSS_FDLDATASPACE		0766
			01	DD	000C6	PUSHL	#1		
			7E	D4	000C8	CLRL	-(SP)		
	67		04	FB	000CA	CALLS	#4, ANL\$FORMAT_LINE		0769
		2C	A6	DD	000CD	PUSHL	MAX_DEPTH		
		00000000G	8F	DD	000D0	PUSHL	#ANLRMSS_FDLDEPTH		0768
			01	DD	000D6	PUSHL	#1		
			7E	D4	000D8	CLRL	-(SP)		
	67		04	FB	000DA	CALLS	#4, ANL\$FORMAT_LINE		

			15	A2	95	000DD	TSTB	21(KP)	0770
				1B	13	000E0	BEQL	3\$	
50		66		02	C7	000E2	DIVL3	#2, DATA_REC_C, R0	0772
		50	10	A6	C0	000E6	ADDL2	DATA_SDR_PTR_C, R0	
		50		66	C6	000EA	DIVL2	DATA_REC_C, R0	0773
				FF	A0	000ED	PUSHAB	-1(R0)	
				00000000G	8F	DD 000F0	PUSHL	#ANLRMSS_FDLDUPSPER	0771
					01	DD 000F6	PUSHL	#1	
					7E	D4 000F8	CLRL	-(SP)	
		67		04	FB	000FA	CALLS	#4, ANL\$FORMAT_LINE	
		03	0000G	CF	B1	000FD	CMPW	ANL\$GW_PROLOG, #3	0774
				1B	12	00102	BNEQ	4\$	
			24	A6	DD	00104	PUSHL	INDEX_REC_LEN_T	0776
7E	24	A6	28	A6	C3	00107	SUBL3	INDEX_COMP_REC_LEN_T, INDEX_REC_LEN_T, -	
								-(SP)	
		68		02	FB	0010D	CALLS	#2, ANL\$PERCENTAGE	
				50	DD	00110	PUSHL	R0	
				00000000G	8F	DD 00112	PUSHL	#ANLRMSS_FDLIDXCOMP	0775
					01	DD 00118	PUSHL	#1	
					7E	D4 0011A	CLRL	-(SP)	
		67		04	FB	0011C	CALLS	#4, ANL\$FORMAT_LINE	
7E	14	A6	18	A6	DD	0011F	ASHL	#9, INDEX_BLK_C, -(SP)	0778
		68		02	FB	00127	PUSHL	INDEX_FILE_T	
				50	DD	0012A	CALLS	#2, ANL\$PERCENTAGE	
				00000000G	8F	DD 0012C	PUSHL	R0	0777
					01	DD 00132	PUSHL	#ANLRMSS_FDLIDXFILL	
					7E	D4 00134	CLRL	#1	
		67		04	FB	00136	CALLS	-(SP)	
			14	A6	DD	00139	CALLS	#4, ANL\$FORMAT_LINE	
				00000000G	8F	DD 0013C	PUSHL	INDEX_BLK_C	0780
					01	DD 00142	PUSHL	#ANLRMSS_FDLIDXSPACE	0779
					7E	D4 00144	CLRL	#1	
		67		04	FB	00146	CALLS	-(SP)	
			20	A6	DD	00149	CALLS	#4, ANL\$FORMAT_LINE	
				00000000G	8F	DD 0014C	PUSHL	INDEX_LEVEL1_REC_C	0782
					01	DD 00152	PUSHL	#ANLRMSS_FDLIDXLTRECS	0781
					7E	D4 00154	PUSHL	#1	
		67		04	FB	00156	CLRL	-(SP)	
50		66		02	C7	00159	CALLS	#4, ANL\$FORMAT_LINE	
		50	04	A6	C0	0015D	DIVL3	#2, DATA_REC_C, R0	0784
		50		66	C7	00161	ADDL2	DATA_REC_LEN_T, R0	
7E				00000000G	8F	DD 00165	DIVL3	DATA_REC_C, R0, -(SP)	0785
					01	DD 0016B	PUSHL	#ANLRMSS_FDLDATALENMEAN	0783
					7E	D4 0016D	PUSHL	#1	
		67		04	FB	0016F	CLRL	-(SP)	
50	1C	A6		02	C7	00172	CALLS	#4, ANL\$FORMAT_LINE	
		50	24	A6	C0	00177	DIVL3	#2, INDEX_REC_C, R0	0787
		50	1C	A6	C7	0017B	ADDL2	INDEX_REC_LEN_T, R0	
7E				00000000G	8F	DD 00180	DIVL3	INDEX_REC_C, R0, -(SP)	0788
					01	DD 00186	PUSHL	#ANLRMSS_FDLIDXLENMEAN	0786
					7E	D4 00188	PUSHL	#1	
		67		04	FB	0018A	CLRL	-(SP)	
0050	8F	00		00	2C	0018D	CALLS	#4, ANL\$FORMAT_LINE	
			F4	A6		00194	MOVCS	#0, (SP), #0, #80, STATISTICS_ACCUMULATORS	0792
				04		00196	RET		0796

RMSSTATS  
V04-000

RMSSTATS - Calculate and Report File Statistics N 5  
ANL\$FDL\_ANALYSIS\_OF\_KEY - Generate FDL Analysis 16-Sep-1984 00:12:49  
14-Sep-1984 11:53:02

VAX-11 Bliss-32 V4.0-742  
[ANALYZ.SRC]RMSSTATS.B32;1

Page 15  
(5)

; Routine Size: 407 bytes, Routine Base: \$CODE\$ + 0037

```

: 294 0797 1 %sbttl 'ANL$AREA_STATISTICS - Print Statistics for Area'
: 295 0798 1 ++
: 296 0799 1 Functional Description:
: 297 0800 1 This routine is called during /STATISTICS mode to print the
: 298 0801 1 statistics for an indexed file area. We are called after the
: 299 0802 1 appropriate statistics have been accumulated via the statistics
: 300 0803 1 callback mechanism described below.
: 301 0804 1
: 302 0805 1 Formal Parameters:
: 303 0806 1 area_id The ID of this area.
: 304 0807 1
: 305 0808 1 Implicit Inputs:
: 306 0809 1 global data
: 307 0810 1
: 308 0811 1 Implicit Outputs:
: 309 0812 1 global data
: 310 0813 1
: 311 0814 1 Returned Value:
: 312 0815 1 none
: 313 0816 1
: 314 0817 1 Side Effects:
: 315 0818 1
: 316 0819 1 --
: 317 0820 1
: 318 0821 1
: 319 0822 2 global routine anl$area_statistics(area_id): novalue = begin
: 320 0823 2
: 321 0824 2
: 322 0825 2 ! We begin with a heading which includes the area ID.
: 323 0826 2
: 324 0827 2 anl$format_skip(0);
: 325 0828 2 anl$format_line(0,0,anlrms$_statarea,.area_id);
: 326 0829 2 anl$format_skip(0);
: 327 0830 2
: 328 0831 2 ! Now we put out the statistic that says how many blocks worth of
: 329 0832 2 ! reclaimed space there is.
: 330 0833 2
: 331 0834 2 anl$format_line(0,1,anlrms$_statrecl,.reclaimed_blk_c);
: 332 0835 2
: 333 0836 2 ! Clear the statistics accumulators for the next area.
: 334 0837 2
: 335 0838 2 ch$fill(%x'00', %allocation(statistics_accumulators),statistics_accumulators);
: 336 0839 2
: 337 0840 2 return;
: 338 0841 2
: 339 0842 1 end;

```

```

                                003C 00000          .ENTRY ANL$AREA_STATISTICS, Save R2,R3,R4,R5
                                7E D4 00002          CLRQ -(SP)
0000G CF                        01 FB 00004          CALLS #1, ANL$FORMAT_SKIP
                                AC DD 00009          PUSHL AREA_ID
                                8F DD 0000C          PUSHL #ANLRMSS$_STATAREA
                                7E 7C 00012          CLRQ -(SP)
: 0822
: 0827
: 0828
:

```

RMSSTATS  
V04-000

RMSSTATS - Calculate and Report File Statistics 16-Sep-1984 00:12:49  
ANL\$AREA\_STATISTICS - Print Statistics for Area 14-Sep-1984 11:53:02

C 6

VAX-11 Bliss-32 V4.0-742  
[ANALYZ.SRC]RMSSTATS.B32;1

Page 17  
(6)

0000G	CF		04	FB	00014	CALLS	#4, ANL\$FORMAT_LINE	:	
			7E	D4	00019	CLRL	-(SP)	:	0829
0000G	CF		01	FB	0001B	CALLS	#1, ANL\$FORMAT_SKIP	:	
		0000'	CF	DD	00020	PUSHL	RECLAIMED_BLK_C	:	0834
		00000000G	8F	DD	00024	PUSHL	#ANLRM\$\$_STATRECL	:	
			01	DD	0002A	PUSHL	#1	:	
			7E	D4	0002C	CLRL	-(SP)	:	
0050	8F	00	0000G	CF	04	FB	0002E	:	
				6E	00	2C	00033	:	0838
					0000'	CF	0003A	:	
					04	0003D	RET	:	0842

; Routine Size: 62 bytes, Routine Base: \$CODE\$ + 01CE

```
0843 1 %sbttl 'ANL$KEY_STATISTICS - Print Statistics for a Key'
0844 1 !++
0845 1 Functional Description:
0846 1 This routine is called during /STATISTICS mode to print the
0847 1 statistics for an indexed file key. We are called after the
0848 1 appropriate statistics have been accumulated via the statistics
0849 1 callback mechanism described below.
0850 1
0851 1 Formal Parameters:
0852 1 key_bsd Address of BSD describing the key.
0853 1
0854 1 Implicit Inputs:
0855 1 global data
0856 1
0857 1 Implicit Outputs:
0858 1 global data
0859 1
0860 1 Returned Value:
0861 1 none
0862 1
0863 1 Side Effects:
0864 1
0865 1 --
0866 1
0867 1
0868 2 global routine anl$key_statistics(key_bsd): novalue = begin
0869 2
0870 2 bind
0871 2 k = .key_bsd: bsd;
0872 2
0873 2 local
0874 2 kp: ref block[,byte],
0875 2 data_bytes: long;
0876 2
0877 2
0878 2 ! We begin with a heading which includes the key ID.
0879 2
0880 2 kp = .k[bsd$l_bufptr] + .k[bsd$l_offset];
0881 2 anl$format_skip(0);
0882 2 anl$format_line(0,0,anlrms$_statkey,.kp[key$b_keyref]);
0883 2 anl$format_skip(0);
0884 2
0885 2 ! If there are no data records in this index because it is uninitialized,
0886 2 ! then let's just put out a comment and quit. Otherwise we'll have a
0887 2 ! lot of trouble with dividing by zero.
0888 2
0889 3 if .data_rec_c eqlu 0 then (
0890 3 anl$format_line(0,1,anlrms$_fdlnorecs);
0891 3 return;
0892 2 );
0893 2
0894 2 ! Now we put out the statistics for the index levels in a logical order.
0895 2 ! Some values only make sense for prologue 3.
0896 2
0897 2 anl$format_line(0,1,anlrms$_statdepth,
0898 2 .max_depth);
0899 2 anl$format_line(0,1,anlrms$_statidxl1recs,
```

```

: 398      0900      2      .index_level1_rec_c);
: 399      0901      2      anl$format_line(0,1,anlrms$_statidxlenmean,
: 400      0902      2      (.index_rec_len_t + .index_rec_c/2) /
: 401      0903      2      .index_rec_c);
: 402      0904      2      anl$format_line(0,1,anlrms$_statidxspace,
: 403      0905      2      .index_blk_c);
: 404      0906      2      anl$format_line(0,1,anlrms$_statidxfill,
: 405      0907      2      anl$percentage(.index_fill_t,.index_blk_c*512));
: 406      0908      2      if .anl$gw_prolog_eqlu plg$c_ver_3 then
: 407      0909      2      anl$format_line(0,1,anlrms$_statidxcomp,
: 408      0910      2      anl$percentage(.index_rec_len_t - .index_comp_rec_len_t,.index_rec_len_t));
: 409      0911      2
: 410      0912      2      ! Now we put out the statistics for the data records in a logical order.
: 411      0913      2      ! Some statistics only make sense for primary or secondary indices.
: 412      0914      2      ! Others make sense only for prologue 3.
: 413      0915      2
: 414      0916      2      anl$format_skip(0);
: 415      0917      2      anl$format_line(0,1,anlrms$_statdatarecs,
: 416      0918      2      .data_rec_c);
: 417      0919      2      if .kp[key$b_keyref] nequ 0 then
: 418      0920      2      anl$format_line(0,1,anlrms$_statdupspcr,
: 419      0921      2      .data_sidr_ptr_c /
: 420      0922      2      .data_rec_c - 1);
: 421      0923      2      anl$format_line(0,1,anlrms$_statdatalenmean,
: 422      0924      2      (.data_rec_len_t + .data_rec_c/2) /
: 423      0925      2      .data_rec_c);
: 424      0926      2      anl$format_line(0,1,anlrms$_statdataspace,
: 425      0927      2      .data_blk_c);
: 426      0928      2      anl$format_line(0,1,anlrms$_statdatafill,
: 427      0929      2      anl$percentage(.data_fill_t,.data_blk_c*512));
: 428      0930      3      if .anl$gw_prolog_eqlu plg$c_ver_3 then (
: 429      0931      3      anl$format_line(0,1,anlrms$_statdatakeycomp,
: 430      0932      3      anl$percentage(.data_rec_c*.kp[key$b_keysz] - .data_comp_key_len_t,.data_rec_c*.kp[key$b_key
: 431      0933      4      if .kp[key$b_keyref] eqlu 0 then (
: 432      0934      4      data_bytes = .data_rec_len_t - .data_rec_c*.kp[key$b_keysz];
: 433      0935      4      anl$format_line(0,1,anlrms$_statdatareccomp,
: 434      0936      4      anl$percentage(.data_bytes - .data_comp_rec_len_t,.data_bytes));
: 435      0937      3      );
: 436      0938      2      );
: 437      0939      2
: 438      0940      2      ! If this is the primary key, then print an indication of the overall
: 439      0941      2      ! space efficiency.
: 440      0942      2
: 441      0943      2      anl$format_skip(0);
: 442      0944      2      if .kp[key$b_keyref] eqlu 0 then
: 443      0945      2      anl$format_line(0,1,anlrms$_statefficiency,
: 444      0946      2      anl$percentage(.data_rec_len_t,.anl$gl_fat[fat$l_hiblk]*512));
: 445      0947      2
: 446      0948      2      ! Clear the statistics accumulators for the next key.
: 447      0949      2
: 448      0950      2      ch$fill(%x'00', %allocation(statistics_accumulators),statistics_accumulators);
: 449      0951      2
: 450      0952      2      return;
: 451      0953      2
: 452      0954      1      end;
```

		03FC 00000		.ENTRY	ANL\$KEY_STATISTICS, Save R2,R3,R4,R5,R6,R7,-;	
		59	0000G	CF 9E 00002	MOVAB ANL\$FORMAT_SKIP, R9	0868
		58	0000V	CF 9E 00007	MOVAB ANL\$PERCENTAGE, R8	
		57	0000G	CF 9E 0000C	MOVAB ANL\$FORMAT_LINE, R7	
		56	0000'	CF 9E 00011	MOVAB DATA_REC_C, R6	
		50	04	AC D0 00016	MOVL KEY_BSD, R0	0871
52	0C	A0	08	A0 C1 0001A	ADDL3 8(R0), 12(R0), KP	0880
				7E D4 00020	CLRL -(SP)	0881
		69		01 FB 00022	CALLS #1, ANL\$FORMAT_SKIP	
		7E	15	A2 9A 00025	MOVZBL 21(KP), -(SP)	0882
			00000000G	8F DD 00029	PUSHL #ANLRMSS_STATKEY	
				7E 7C 0002F	CLRL -(SP)	
		67		04 FB 00031	CALLS #4, ANL\$FORMAT_LINE	
				7E D4 00034	CLRL -(SP)	0883
		69		01 FB 00036	CALLS #1, ANL\$FORMAT_SKIP	
				66 D5 00039	TSTL DATA_REC_C	0889
				0E 12 0003B	BNEQ 1\$	
			00000000G	8F DD 0003D	PUSHL #ANLRMSS_FDLNORECS	0890
				01 DD 00043	PUSHL #1	
				7E D4 00045	CLRL -(SP)	
		67		03 FB 00047	CALLS #3, ANL\$FORMAT_LINE	
				04 0004A	RET	0889
			2C	A6 DD 0004B	PUSHL MAX_DEPTH	0898
			00000000G	8F DD 0004E	PUSHL #ANLRMSS_STATDEPTH	0897
				01 DD 00054	PUSHL #1	
				7E D4 00056	CLRL -(SP)	
		67		04 FB 00058	CALLS #4, ANL\$FORMAT_LINE	
			20	A6 DD 0005B	PUSHL INDEX_LEVEL1_REC_C	0900
			00000000G	8F DD 0005E	PUSHL #ANLRMSS_STATIDX1RECS	0899
				01 DD 00064	PUSHL #1	
				7E D4 00066	CLRL -(SP)	
		67		04 FB 00068	CALLS #4, ANL\$FORMAT_LINE	
50	1C	A6		02 C7 0006B	DIVL3 #2, INDEX_REC_C, R0	0902
		50	24	A6 C0 00070	ADDL2 INDEX_REC_LEN_T, R0	
7E		50	1C	A6 C7 00074	DIVL3 INDEX_REC_C, R0, -(SP)	0903
			00000000G	8F DD 00079	PUSHL #ANLRMSS_STATIDXLENMEAN	0901
				01 DD 0007F	PUSHL #1	
				7E D4 00081	CLRL -(SP)	
		67		04 FB 00083	CALLS #4, ANL\$FORMAT_LINE	
			14	A6 DD 00086	PUSHL INDEX_BLK_C	0905
			00000000G	8F DD 00089	PUSHL #ANLRMSS_STATIDXSPACE	0904
				01 DD 0008F	PUSHL #1	
				7E D4 00091	CLRL -(SP)	
		67		04 FB 00093	CALLS #4, ANL\$FORMAT_LINE	
50	14	A6		09 78 00096	ASHL #9, INDEX_BLK_C, -(SP)	0907
			18	A6 DD 0009B	PUSHL INDEX_FILE_T	
		68		02 FB 0009E	CALLS #2, ANL\$PERCENTAGE	
				50 DD 000A1	PUSHL R0	
			00000000G	8F DD 000A3	PUSHL #ANLRMSS_STATIDXFILL	0906
				01 DD 000A9	PUSHL #1	
				7E D4 000AB	CLRL -(SP)	
		67		04 FB 000AD	CALLS #4, ANL\$FORMAT_LINE	
		03	0000G	CF B1 000B0	CMPW ANL\$GW_PROLOG, #3	0908

7E	24	A6	24 28	1B A6 A6	12 DD C3	000B5 000B7 000BA	BNEQ PUSHL SUBL3	2\$ INDEX_REC_LEN_T INDEX_COMP_REC_LEN_T, INDEX_REC_LEN_T, - -(SP)	0910
			68		02	FB 000C0	CALLS	#2, ANL\$PERCENTAGE	
					50	DD 000C3	PUSHL	R0	0909
			00000000G		8F	DD 000C5	PUSHL	#ANLRMSS_STATIDXCOMP	
					01	DD 000CB	PUSHL	#1	
					7E	D4 000CD	CLRL	-(SP)	
			67		04	FB 000CF	CALLS	#4, ANL\$FORMAT_LINE	0916
					7E	D4 000D2	CLRL	-(SP)	
			69		01	FB 000D4	CALLS	#1, ANL\$FORMAT_SKIP	0918
					66	DD 000D7	PUSHL	DATA_REC_C	0917
			00000000G		8F	DD 000D9	PUSHL	#ANLRMSS_STATDATARECS	
					01	DD 000DF	PUSHL	#1	
					7E	D4 000E1	CLRL	-(SP)	
			67		04	FB 000E3	CALLS	#4, ANL\$FORMAT_LINE	0919
					A2	95 000E6	TSTB	21(KP)	
			15		15	13 000E9	BEQL	3\$	0922
					66	C7 000EB	DIVL3	DATA_REC_C, DATA_SIDR_PTR_C, R0	0920
					A0	9F 000F0	PUSHAB	-1(R0)	
			00000000G		8F	DD 000F3	PUSHL	#ANLRMSS_STATDUPSPER	
					01	DD 000F9	PUSHL	#1	
					7E	D4 000FB	CLRL	-(SP)	
			67		04	FB 000FD	CALLS	#4, ANL\$FORMAT_LINE	0924
			50		02	C7 00100	DIVL3	#2, DATA_REC_C, R0	
					A6	C0 00104	ADDL2	DATA_REC_LEN_T, R0	0925
			67		66	C7 00108	DIVL3	DATA_REC_C, R0, -(SP)	0923
					8F	DD 0010C	PUSHL	#ANLRMSS_STATDATALENMEAN	
			00000000G		01	DD 00112	PUSHL	#1	
					7E	D4 00114	CLRL	-(SP)	
			67		04	FB 00116	CALLS	#4, ANL\$FORMAT_LINE	0927
					A6	DD 00119	PUSHL	DATA_BLK_C	0926
			00000000G		8F	DD 0011C	PUSHL	#ANLRMSS_STATDATASPACE	
					01	DD 00122	PUSHL	#1	
					7E	D4 00124	CLRL	-(SP)	
			67		04	FB 00126	CALLS	#4, ANL\$FORMAT_LINE	0929
					09	78 00129	ASHL	#9, DATA_BLK_C, -(SP)	
			67		A6	DD 0012E	PUSHL	DATA_FILE_T	
					02	FB 00131	CALLS	#2, ANL\$PERCENTAGE	
					50	DD 00134	PUSHL	R0	
			00000000G		8F	DD 00136	PUSHL	#ANLRMSS_STATDATAFILL	0928
					01	DD 0013C	PUSHL	#1	
					7E	D4 0013E	CLRL	-(SP)	
			67		04	FB 00140	CALLS	#4, ANL\$FORMAT_LINE	0930
					CF	B1 00143	CMPW	ANL\$GW_PROLOG, #3	
			03		4A	12 00148	BNEQ	4\$	
					A2	9A 0014A	MOVZBL	20(KP), R0	0932
			50		66	C4 0014E	MULL2	DATA_REC_C, R0	
					50	DD 00151	PUSHL	R0	
			50		A6	C3 00153	SUBL3	DATA_COMP_KEY_LEN_T, R0, -(SP)	0931
					02	FB 00158	CALLS	#2, ANL\$PERCENTAGE	
			67		50	DD 0015B	PUSHL	R0	
					8F	DD 0015D	PUSHL	#ANLRMSS_STATDATAKEYCOMP	
			00000000G		01	DD 00163	PUSHL	#1	
					7E	D4 00165	CLRL	-(SP)	
			67		04	FB 00167	CALLS	#4, ANL\$FORMAT_LINE	

			15	A2	95	0016A	TSTB	21(KP)	:	0933
				25	12	0016D	BNEQ	4\$	:	
		50	14	A2	9A	0016F	MOVZBL	20(KP), R0	:	0934
		50		66	C4	00173	MULL2	DATA_REC_C, R0	:	
50	04	A6		50	C3	00176	SUBL3	R0, DATA_REC_LEN_T, DATA_BYTES	:	
				50	DD	0017B	PUSHL	DATA_BYTES	:	0936
7E		50	0C	A6	C3	0017D	SUBL3	DATA_COMP_REC_LEN_T, DATA_BYTES, -(SP)	:	
		68		02	FB	00182	CALLS	#2, ANL\$PERCENTAGE	:	
				50	DD	00185	PUSHL	R0	:	
			00000000G	8F	DD	00187	PUSHL	#ANLRM\$\$_STATDATARECCOMP	:	0935
				01	DD	0018D	PUSHL	#1	:	
				7E	D4	0018F	CLRL	-(SP)	:	
		67		04	FB	00191	CALLS	#4, ANL\$FORMAT_LINE	:	
				7E	D4	00194	CLRL	-(SP)	:	0943
		69		01	FB	00196	CALLS	#1, ANL\$FORMAT_SKIP	:	
			15	A2	95	00199	TSTB	21(KP)	:	0944
				1F	12	0019C	BNEQ	5\$	:	
		50	0000G	CF	D0	0019E	MOVL	ANL\$GL FAT, R0	:	0946
7E	04	A0		09	78	001A3	ASHL	#9, 4(R0), -(SP)	:	
			04	A6	DD	001A8	PUSHL	DATA_REC_LEN_T	:	
		68		02	FB	001AB	CALLS	#2, ANL\$PERCENTAGE	:	
				50	DD	001AE	PUSHL	R0	:	
			00000000G	8F	DD	001B0	PUSHL	#ANLRM\$\$_STATEFFICIENCY	:	0945
				01	DD	001B6	PUSHL	#1	:	
				7E	D4	001B8	CLRL	-(SP)	:	
		67		04	FB	001BA	CALLS	#4, ANL\$FORMAT_LINE	:	
0050	8F	00		00	2C	001BD	MOVCS	#0, (SP), #0, #80, STATISTICS_ACCUMULATORS	:	0950
		6E		00	2C	001BD			:	
			F4	A6		001C4			:	
				04		001C6	RET		:	0954

; Routine Size: 455 bytes, Routine Base: \$CODE\$ + 020C

```

: 454 0955 1 %sbttl 'ANL$PERCENTAGE - Calculate a Percentage'
: 455 0956 1 ++
: 456 0957 1 Functional Description:
: 457 0958 1 This routine is called to calculate a percentage.
: 458 0959 1
: 459 0960 1 Formal Parameters:
: 460 0961 1 dividend The dividend in the percentage.
: 461 0962 1 divisor The divisor in the percentage.
: 462 0963 1
: 463 0964 1 Implicit Inputs:
: 464 0965 1 global data
: 465 0966 1
: 466 0967 1 Implicit Outputs:
: 467 0968 1 global data
: 468 0969 1
: 469 0970 1 Returned Value:
: 470 0971 1 The percentage.
: 471 0972 1
: 472 0973 1 Side Effects:
: 473 0974 1
: 474 0975 1 --
: 475 0976 1
: 476 0977 1
: 477 0978 2 global routine anl$percentage(dividend,divisor) = begin
: 478 0979 2
: 479 0980 2 local
: 480 0981 2 quadword: block[8,byte],
: 481 0982 2 quotient: signed long, remainder: signed long;
: 482 0983 2
: 483 0984 2 builtin
: 484 0985 2 emul, ediv;
: 485 0986 2
: 486 0987 2
: 487 0988 2 ! If the divisor is zero, then just return zero.
: 488 0989 2
: 489 0990 2 if .divisor eqlu 0 then
: 490 0991 2 return 0;
: 491 0992 2
: 492 0993 2 ! Calculate the percentage using quadword arithmetic.
: 493 0994 2
: 494 0995 2 emul(dividend,%ref(100),%ref(0),quadword);
: 495 0996 2 ediv(divisor,quadword,quotient,remainder);
: 496 0997 2
: 497 0998 2 return .quotient;
: 498 0999 2
: 499 1000 1 end;

```

				0000 0000	.ENTRY	ANL\$PERCENTAGE, Save nothing	: 0978
	5E		08	08 C2 00002	SUBL2	#8, SP	: 0990
			08	AC D5 00005	TSTL	DIVISOR	: 0995
				14 13 00008	BEQL	1\$	: 0996
6E	00 00000064	8F	04	AC 7A 0000A	EMUL	DIVIDEND, #100, #0, QUADWORD	: 0995
50	51	6E	08	AC 7B 00014	EDIV	DIVISOR, QUADWORD, QUOTIENT, REMAINDER	: 0996

RMSSTATS  
V04-000

RMSSTATS - Calculate and Report File Statistics  
ANL\$PERCENTAGE - Calculate a Percentage

J 6  
16-Sep-1984 00:12:49  
14-Sep-1984 11:53:02

VAX-11 Bliss-32 V4.0-742  
[ANALYZ.SRC]RMSSTATS.B32;1

Page 24  
(8)

50

51 D0 0001A  
04 0001D  
50 D4 0001E 1\$:  
04 00020

MOVL QUOTIENT, R0  
RET  
CLRL R0  
RET

: 0998  
:  
: 1000  
:

; Routine Size: 33 bytes, Routine Base: \$CODE\$ + 03D3

STATISTICS CALLBACK ROUTINES

```
: 501 1001 1 |
: 502 1002 1 |
: 503 1003 1 | The method by which we collect the statistics for an RMS file is known
: 504 1004 1 | as statistics callback routines. These are routines in this module
: 505 1005 1 | which are "called back" by the various low-level structure analysis
: 506 1006 1 | routines. The analysis routines pass enough information so that the
: 507 1007 1 | callback routines can accumulate raw statistical information. After the
: 508 1008 1 | analysis is finished, one of the statistics formatting routines
: 509 1009 1 | (ANL$STATISTICS_FDL or ANL$STATISTICS_MODE) can calculate the final
: 510 1010 1 | statistics and format them.
: 511 1011 1 |
: 512 1012 1 | When an analysis routine wants to call back a statistics routine, it
: 513 1013 1 | uses the STATISTICS_CALLBACK macro. This macro takes an arbitrary
: 514 1014 1 | number of statements as its argument, and causes them to be executed
: 515 1015 1 | only if the current mode does statistics (/FDL or /STATISTICS).
```

```

: 517 1016 1 %sbttl 'ANL$BUCKET_CALLBACK - Statistics Callback for Buckets'
: 518 1017 1 ++
: 519 1018 1 Functional Description:
: 520 1019 1 This is the statistics callback routine for the analysis of a bucket.
: 521 1020 1 We keep track of information about the buckets.
: 522 1021 1
: 523 1022 1 Formal Parameters:
: 524 1023 1 level The level of this bucket.
: 525 1024 1 blocks Number of blocks in the bucket.
: 526 1025 1 fill Fill count for this bucket.
: 527 1026 1
: 528 1027 1 Implicit Inputs:
: 529 1028 1 global data
: 530 1029 1
: 531 1030 1 Implicit Outputs:
: 532 1031 1 global data
: 533 1032 1
: 534 1033 1 Returned Value:
: 535 1034 1 none
: 536 1035 1
: 537 1036 1 Side Effects:
: 538 1037 1
: 539 1038 1 --
: 540 1039 1
: 541 1040 1
: 542 1041 2 global routine anl$bucket_callback(level,blocks,fill): novalue = begin
: 543 1042 2
: 544 1043 2
: 545 1044 2 ! We need to keep track of the space occupied and the data fill count.
: 546 1045 2 ! It's done separately for data and index.
: 547 1046 2
: 548 1047 3 if .level eglu 0 then (
: 549 1048 3 data_blk_c = .data_blk_c + .blocks;
: 550 1049 3 data_fill_t = .data_fill_t + .fill;
: 551 1050 3 ) else (
: 552 1051 3 index_blk_c = .index_blk_c + .blocks;
: 553 1052 3 index_fill_t = .index_fill_t + .fill;
: 554 1053 2 );
: 555 1054 2
: 556 1055 2 return;
: 557 1056 2
: 558 1057 1 end;

```

			0000 0000	.ENTRY	ANL\$BUCKET_CALLBACK, Save nothing	: 1041
		04	AC D5 00002	TSTL	LEVEL	: 1047
			0D 12 00005	BNEQ	1\$	:
0000*	CF	08	AC C0 00007	ADDL2	BLOCKS, DATA_BLK_C	: 1048
0000*	CF	0C	AC C0 0000D	ADDL2	FILL, DATA_FILL_T	: 1049
			04 00013	RET		: 1047
0000*	CF	08	AC C0 00014 1\$:	ADDL2	BLOCKS, INDEX_BLK_C	: 1051
0000*	CF	0C	AC C0 0001A	ADDL2	FILL, INDEX_FILL_T	: 1052
			04 00020	RET		: 1057

RMSSTATS  
V04-000

RMSSTATS - Calculate and Report File Statistics M 6 16-Sep-1984 00:12:49  
ANL\$BUCKET\_CALLBACK - Statistics Callback for B 14-Sep-1984 11:53:02

VAX-11 Bliss-32 V4.0-742  
[ANALYZ.SRC]RMSSTATS.B32;1

Page 27  
(10)

; Routine Size: 33 bytes, Routine Base: \$CODE\$ + 03F4

```

: 560      1058 1 %sbttl 'ANL$RECLAIMED_BUCKET_CALLBACK - Statistics Callback for Reclaimed'
: 561      1059 1 ++
: 562      1060 1 Functional Description:
: 563      1061 1 This is the statistics callback routine for reclaimed buckets
: 564      1062 1 hanging off an area descriptor.
: 565      1063 1
: 566      1064 1 Formal Parameters:
: 567      1065 1 blocks          Number of blocks in the bucket.
: 568      1066 1
: 569      1067 1 Implicit Inputs:
: 570      1068 1 global data
: 571      1069 1
: 572      1070 1 Implicit Outputs:
: 573      1071 1 global data
: 574      1072 1
: 575      1073 1 Returned Value:
: 576      1074 1 none
: 577      1075 1
: 578      1076 1 Side Effects:
: 579      1077 1
: 580      1078 1 --
: 581      1079 1
: 582      1080 1
: 583      1081 2 global routine anl$reclaimed_bucket_callback(blocks): novalue = begin
: 584      1082 2
: 585      1083 2
: 586      1084 2 ! All we need to do is keep track of the number of reclaimed blocks.
: 587      1085 2
: 588      1086 2 reclaimed_blk_c = .reclaimed_blk_c + .blocks;
: 589      1087 2
: 590      1088 2 return;
: 591      1089 2
: 592      1090 1 end;

```

```

0000* CF      04      AC      C0 00002      .ENTRY ANL$RECLAIMED_BUCKET_CALLBACK, Save nothing ; 1081
04 00008      RET      BLOCKS, RECLAIMED_BLK_C ; 1086
; 1090

```

; Routine Size: 9 bytes, Routine Base: \$CODE\$ + 0415

```

: 594 1091 1 %sbttl 'ANL$INDEX_CALLBACK - Statistics Callback for Index Record'
: 595 1092 1 ++
: 596 1093 1 Functional Description:
: 597 1094 1 This is the statistics callback routine for an index record. We
: 598 1095 1 accumulate various information about the index.
: 599 1096 1
: 600 1097 1 Formal Parameters:
: 601 1098 1 level Level of the index bucket.
: 602 1099 1 record_len Nominal length of the entire index record.
: 603 1100 1 compressed_len Compressed length of the entire index record.
: 604 1101 1
: 605 1102 1 Implicit Inputs:
: 606 1103 1 global data
: 607 1104 1
: 608 1105 1 Implicit Outputs:
: 609 1106 1 global data
: 610 1107 1
: 611 1108 1 Returned Value:
: 612 1109 1
: 613 1110 1
: 614 1111 1 Side Effects:
: 615 1112 1
: 616 1113 1 --
: 617 1114 1
: 618 1115 1
: 619 1116 2 global routine anl$index_callback(level,record_len,compressed_len): novalue = begin
: 620 1117 2
: 621 1118 2
: 622 1119 2 ! Count this index record. Keep a separate count of level 1 records.
: 623 1120 2
: 624 1121 2 increment (index_rec_c);
: 625 1122 2 if .level eqlu 1 then
: 626 1123 2 increment (index_level1_rec_c);
: 627 1124 2
: 628 1125 2 ! We have to keep track of the total nominal record length, and the total
: 629 1126 2 ! compressed record length.
: 630 1127 2
: 631 1128 2 index_rec_len_t = .index_rec_len_t + .record_len;
: 632 1129 2 index_comp_rec_len_t = .index_comp_rec_len_t + .compressed_len;
: 633 1130 2
: 634 1131 2 ! Now we must keep track of the maximum index depth.
: 635 1132 2
: 636 1133 2 max_depth = maxu(.max_depth,.level);
: 637 1134 2
: 638 1135 2 return;
: 639 1136 2
: 640 1137 1 end;
  
```

```

52 0000 0004 0000 .ENTRY ANL$INDEX_CALLBACK, Save R2 : 1116
CF 9E 0002 MOVAB MAX_DEPTH, R2 : 1121
F0 A2 D6 0007 INCL INDEX_REC_C : 1122
01 04 AC D1 0000A CMPL LEVEL, #1
03 12 0000E BNEQ 1$
  
```

RMSSTATS  
V04-000

C 7

RMSSTATS - Calculate and Report File Statistics 16-Sep-1984 00:12:49  
ANL\$INDEX\_CALLBACK - Statistics Callback for In 14-Sep-1984 11:53:02

VAX-11 Bliss-32 V4.0-742  
[ANALYZ.SRC]RMSSTATS.B32;1

Page 30  
(12)

F8	A2	F4	A2	D6	00010	INCL	INDEX_LEVEL1_REC_C	:	1123
FC	A2	08	AC	C0	00013	ADDL2	RECORD_LEN, INDEX_REC_LEN_T	:	1128
	50	0C	AC	C0	00018	ADDL2	COMPRESSED_LEN, INDEX_COMP_REC_LEN_T	:	1129
04	AC		62	D0	0001D	MOVL	MAX_DEPTH, RO	:	1133
			50	D1	00020	CMPL	RO, LEVEL	:	
			04	1E	00024	BGEQU	2\$	:	
	50	04	AC	D0	00026	MOVL	LEVEL, RO	:	
	62		50	D0	0002A	MOVL	RO, MAX_DEPTH	:	
			04	0002D	2\$:	RET		:	1137

; Routine Size: 46 bytes, Routine Base: \$CODE\$ + 041E

```

: 642 1138 1 %sbttl 'ANL$DATA_CALLBACK - Statistics Callback for Data Records'
: 643 1139 1 ++
: 644 1140 1 Functional Description:
: 645 1141 1 This is the statistics callback routine for data records. We
: 646 1142 1 accumulate various information about the records.
: 647 1143 1
: 648 1144 1 Formal Parameters:
: 649 1145 1 record_len Nominal length of this data record.
: 650 1146 1 compressed_key_len Compressed length of the key.
: 651 1147 1 compressed_data_len Compressed length of the data bytes.
: 652 1148 1 sidr_pointers Count of pointers in this SIDR.
: 653 1149 1
: 654 1150 1 Implicit Inputs:
: 655 1151 1 global data
: 656 1152 1
: 657 1153 1 Implicit Outputs:
: 658 1154 1 global data
: 659 1155 1
: 660 1156 1 Returned Value:
: 661 1157 1 none
: 662 1158 1
: 663 1159 1 Side Effects:
: 664 1160 1
: 665 1161 1 --
: 666 1162 1
: 667 1163 1
: 668 1164 2 global routine anl$data_callback(record_len,compressed_key_len,compressed_data_len,sidr_pointers): novalue =
: 669 1165 2
: 670 1166 2
: 671 1167 2 ! Count this data record.
: 672 1168 2
: 673 1169 2 increment (data_rec_c);
: 674 1170 2
: 675 1171 2 ! Keep track of the total nominal record length, total compressed key length,
: 676 1172 2 ! and total compressed data length.
: 677 1173 2
: 678 1174 2 data_rec_len_t = .data_rec_len_t + .record_len;
: 679 1175 2 data_comp_key_len_t = .data_comp_key_len_t + .compressed_key_len;
: 680 1176 2 data_comp_rec_len_t = .data_comp_rec_len_t + .compressed_data_len;
: 681 1177 2
: 682 1178 2 ! For secondary data, keep track of the total number of SIDR pointers.
: 683 1179 2
: 684 1180 2 data_sidr_ptr_c = .data_sidr_ptr_c + .sidr_pointers;
: 685 1181 2
: 686 1182 2 return;
: 687 1183 2
: 688 1184 1 end;

```

```

          52      0000'  CF  0004 00000      .ENTRY  ANL$DATA_CALLBACK, Save R2      : 1164
          04  A2      04  AC  62  9E 00002      MOVAB  DATA_REC_C, R2      :
          08  A2      08  AC  62  D6 00007      INCL  DATA_REC_C      : 1169
          08  A2      08  AC  C0 00009      ADDL2 RECORD_LEN, DATA_REC_LEN_T      : 1174
          08  A2      08  AC  C0 0000E      ADDL2 COMPRESSED_KEY_LEN, DATA_COMP_KEY_LEN_T      : 1175

```

OC	A2	OC	AC	CO	00013	ADD	COMPRESSED DATA LEN, DATA_COMP_REC_LEN_T	:	1176
10	A2	10	AC	CO	00018	ADD	SIDR_POINTERS, DATA_SIDR_PTR_C	:	1180
				04	0001D	RET		:	1184

: Routine Size: 30 bytes, Routine Base: \$CODE\$ + 044C

: 689 1185 1  
: 690 1186 0 end eludom

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	80	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	1130	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	14	0	1000	00:01.8

COMMAND QUALIFIERS

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

: Size: 1130 code + 80 data bytes  
 : Run Time: 00:22.1  
 : Elapsed Time: 01:04.5  
 : Lines/CPU Min: 3222  
 : Lexemes/CPU-Min: 11831  
 : Memory Used: 222 pages  
 : Compilation Complete

0009 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

A grid of 10 columns and 10 rows of terminal windows. Each window contains a different type of data or report. The windows are arranged in a regular grid pattern across the page.

System Information									
System Information									
System Information									
System Information									
System Information									
System Information									
System Information									
System Information									
System Information									
System Information									

BACKDEF  
SOL

COMMON  
REQ

BACKUP

STABACKUP  
MAP

BACKUP  
MAP

RMSREPORT  
LIS

RMSSTATS  
LIS

STABACKUP  
MAP