```
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB        AAAAAAAAA        CCCCCCCCCCC  KKK         KKK  UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBB         BBB  AAA         AAA  CCC            KKK       KKK    UUU          UUU  PPP         PPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBBBBBBBBBBB     AAA         AAA  CCC            KKKKKKKKK        UUU          UUU  PPPPPPPPPPPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAAAAAAAAAAAAAA  CCC            KKK       KKK    UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP
BBB         BBB  AAA         AAA  CCC            KKK         KKK  UUU          UUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK         KKK  UUUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK         KKK  UUUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK         KKK  UUUUUUUUUUUUUUUU  PPP
BBBBBBBBBBBB     AAA         AAA     CCCCCCCCCCC  KKK         KKK  UUUUUUUUUUUUUUUU  PPP
```

FILEID**STAINIT

```
  SSSSSSSS  TTTTTTTTTT    AAAAAA     IIIIII    NN      NN    IIIIII    TTTTTTTTTT
  SSSSSSSS  TTTTTTTTTT    AAAAAA     IIIIII    NN      NN    IIIIII    TTTTTTTTTT
SS              TT       AA    AA      II     NN      NN      II        TT
SS              TT       AA    AA      II     NN      NN      II        TT
SS              TT       AA    AA      II     NNNN    NN      II        TT
SS              TT       AA    AA      II     NNNN    NN      II        TT
  SSSSSS        TT       AA    AA      II     NN  NN  NN      II        TT
  SSSSSS        TT       AA    AA      II     NN  NN  NN      II        TT
        SS      TT       AAAAAAAAAA      II     NN    NNNN      II        TT
        SS      TT       AAAAAAAAAA      II     NN    NNNN      II        TT
        SS      TT       AA    AA      II     NN      NN      II        TT
        SS      TT       AA    AA      II     NN      NN      II        TT
SSSSSSSS        TT       AA    AA    IIIIII   NN      NN    IIIIII      TT
SSSSSSSS        TT       AA    AA    IIIIII   NN      NN    IIIIII      TT

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

```
;     1        0001  0 MODULE STAINIT  (%TITLE 'Standalone BACKUP initialization'
;     2        0002  0                  IDENT = 'V04-000'
;     3        0003  0                  ) =
;     4        0004  1 BEGIN
;     5        0005  1
;     6        0006  1 !
;     7        0007  1 !**********************************************************************************
;     8        0008  1 !*                                                                                *
;     9        0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                       *
;    10        0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                         *
;    11        0011  1 !*  ALL RIGHTS RESERVED.                                                           *
;    12        0012  1 !*                                                                                *
;    13        0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED         *
;    14        0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
;    15        0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER        *
;    16        0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY        *
;    17        0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY        *
;    18        0018  1 !*  TRANSFERRED.                                                                   *
;    19        0019  1 !*                                                                                *
;    20        0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE         *
;    21        0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT         *
;    22        0022  1 !*  CORPORATION.                                                                   *
;    23        0023  1 !*                                                                                *
;    24        0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS        *
;    25        0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                        *
;    26        0026  1 !*                                                                                *
;    27        0027  1 !*                                                                                *
;    28        0028  1 !**********************************************************************************
;    29        0029  1 !
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 ! FACILITY:
;    33        0033  1 !       Backup/Restore
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !       This module contains the routines that initialize the standalone
;    37        0037  1 !       BACKUP.
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:
;    40        0040  1 !       VAX/VMS user, executive, kernel mode.
;    41        0041  1 !--
;    42        0042  1 !
;    43        0043  1 ! AUTHOR: M. Jack, CREATION DATE: 06-Jan-1981
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       V03-003 CWH3003         CW Hobbs,               29-Oct-1983
;    48        0048  1 !               Change image name to STANDALON.EXE to match change to
;    49        0049  1 !               standalone VMS packaging.
;    50        0050  1 !
;    51        0051  1 !       V03-002 ACG53600        Andrew C. Goldstein,    10-Feb-1983  19:13
;    52        0052  1 !               Output ident message at startup. Condition disabling of
;    53        0053  1 !               bugcheck code on DUMPBUG SYSGEN parameter.
;    54        0054  1 !
;    55        0055  1 !       V03-001 MLJ0085         Martin L. Jack, 30-Mar-1982  12:59
;    56        0056  1 !               Copy a small routine over EXE$BUG_CHECK that types out a
;    57        0057  1 !               console message "Bugcheck" if a bugcheck occurs.  Since the
```

```
;   58          0058  1 !                 console media containing the executive is not resident, the
;   59          0059  1 !                 bugcheck code cannot be loaded.
;   60          0060  1 !
;   61          0061  1 !**
```

```
;    63          0062  1 REQUIRE 'SRC$:COMMON';
;    64          1168  1 LIBRARY 'SYS$LIBRARY:LIB';
;    65          1169  1
;    66          1170  1
;    67          1171  1 LINKAGE
;    68          1172  1         JSB=            JSB: NOPRESERVE(2,3,4,5,6,7,8,9,10,11),
;    69          1173  1         JSB_R0=         JSB(REGISTER=0): PRESERVE(0,1,2,3,4,5,6,7,8,9,10,11),
;    70          1174  1         JSB_PRESERVE=   JSB:;
;    71          1175  1
;    72          1176  1
;    73          1177  1 FORWARD ROUTINE
;    74          1178  1         STA_INIT:       NOVALUE,            ! Driver for initialization
;    75          1179  1         STA_BUGCHECK:   JSB NOVALUE,        ! Routine copied over EXE$BUG_CHECK
;    76          1180  1         STA_BUG_INSTALL,                    ! Install bugcheck code
;    77          1181  1         STA_HANDLER,                        ! Last-chance handler
;    78          1182  1         PUTMSG_ACTRTN,                      ! $PUTMSG action routine for handler
;    79          1183  1         STA_RESTART:    NOVALUE;            ! Restart standalone version
;    80          1184  1
;    81          1185  1
;    82          1186  1 EXTERNAL ROUTINE
;    83          1187  1         BOO$ACTIMAGE,                       ! Reactivate image
;    84          1188  1         CLI$DCL_PARSE:  ADDRESSING_MODE(GENERAL),
;    85          1189  1                                             ! Stand-alone command parser
;    86          1190  1         LIB$GET_COMMAND:ADDRESSING_MODE(GENERAL),
;    87          1191  1                                             ! Stand-alone get from SYS$COMMAND
;    88          1192  1         CON$PUTCHAR:ADDRESSING_MODE(GENERAL) JSB_R0 NOVALUE,
;    89          1193  1                                             ! Put a character out to the console
;    90          1194  1                                             ! terminal
;    91          1195  1         CON$OWNCTY:ADDRESSING_MODE(GENERAL) JSB_PRESERVE NOVALUE;
;    92          1196  1
;    93          1197  1
;    94          1198  1 EXTERNAL
;    95          1199  1         EXE$GL_FLAGS:   BITVECTOR       ADDRESSING_MODE(GENERAL),
;    96          1200  1                                         ! Executive flags longword
;    97          1201  1         VERSION_STRING  : VECTOR [,BYTE], ! BACKUP version string
;    98          1202  1         BACKUPCMD;                      ! Stand-alone command tables
;    99          1203  1
;   100          1204  1
;   101          1205  1 EXTERNAL LITERAL
;   102          1206  1         BACKUP$_IDENT,
;   103          1207  1         EXE$V_INIT:     UNSIGNED(6);    ! True if RMS and ACP are active
;   104          1208  1
;   105          1209  1
;   106          1210  1 G$DEFINE();                    ! Define global area
```

```
1 %SBTTL 'STA_INIT - Stand-alone BACKUP initialization'
1 GLOBAL ROUTINE STA_INIT: NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is the driver for initialization of the stand-alone
1 !       BACKUP.
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       NONE
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       NONE
1 !
1 !--
1
2 BEGIN
2 LOCAL
2       BUFFER:         VECTOR[132,BYTE],       ! Command buffer
2       DESC:           BBLOCK[8];              ! Local descriptor
2 BUILTIN
2       FP;
2 MAP
2       FP:             REF BBLOCK;
2
2
2 ! Establish the general handler.  Since this routine is called by the main
2 ! routine, this code will establish it as a stack handler in that routine.
2 !
2 .FP[SF$L_SAVE_FP] = STA_HANDLER;
2
2
2 ! If we are really running standalone, copy our own routine over the exec's
2 ! bugcheck code.
2 !
2 IF NOT .EXE$GL_FLAGS[EXE$V_INIT]
2 THEN
2     $CMKRNL(ROUTIN=STA_BUG_INSTALL);
2
2
2 ! Output the ident message.
2 !
2
2 SIGNAL (BACKUP$_IDENT, 3, %CHARCOUNT (BACKUP$VERSION), VERSION_STRING, 0);
```

```
;  165          1268  2
;  166          1269  2
;  167          1270  2 ! Get the command.
;  168          1271  2 !
;  169          1272  2 COM_FLAGS[COM_STANDALONE] = TRUE;
;  170          1273  2 DO
;  171          1274  3     BEGIN
;  172          1275  3     DESC[DSC$W_LENGTH] = 132;
;  173          1276  3     DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  174          1277  3     DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  175          1278  3     DESC[DSC$A_POINTER] = BUFFER;
;  176          1279  3     LIB$GET_COMMAND(DESC, $DESCRIPTOR(%CHAR(%O'012'), '$ '), DESC);
;  177          1280  3     END
;  178          1281  2 UNTIL
;  179          1282  3     BEGIN
;  180          1283  3     IF .DESC[DSC$W_LENGTH] EQL 0
;  181          1284  3         THEN FALSE
;  182          1285  3         ELSE CLI$DCL_PARSE(DESC, BACKUPCMD)
;  183          1286  2     END;
;  184          1287  1 END;


                                                        .TITLE  STAINIT Standalone BACKUP initialization
                                                        .IDENT  \V04-000\

                                                        .PSECT  COMMON,NOEXE,  OVR,2

                                                00000 GLOBAL_BASE:
                                                                .BLKB   0
                                                00000 FREE_LIST:
                                                                .BLKB   8
                                                00008 INPUT_WAIT:
                                                                .BLKB   8
                                                00010 REREAD_WAIT:
                                                                .BLKB   8
                                                00018 OUTPUT_WAIT:
                                                                .BLKB   8
                                                00020 JPI_UIC:  .BLKB   4
                                                00024 JPI_USERNAME:
                                                                .BLKB   12
                                                00030 JPI_DATE:
                                                                .BLKB   8
                                                00038 JPI_NODE_DESC:
                                                                .BLKB   8
                                                00040 JPI_CURPRIV:
                                                                .BLKB   8
                                                00048 SYI_VERSION:
                                                                .BLKB   4
                                                0004C SYI_SID:  .BLKB   4
                                                00050 RWSV_HOLD_LIST:
                                                                .BLKB   8
                                                00058 RWSV_CRC16:
                                                                .BLKB   64
                                                00098 RWSV_AUTODIN:
                                                                .BLKB   64
                                                000D8 RWSV_FILESET_ID:
                                                                .BLKB   8
```

```
000E0 RWSV_VOLUME_ID:
                .BLKB   12
000EC RWSV_VOL_NUMBER:
                .BLKB   2
000EE RWSV_SEG_NUMBER:
                .BLKB   2
000F0 RWSV_FILE_NUMBER:
                .BLKB   4
000F4 RWSV_SAVE_QUAL:
                .BLKB   4
000F8 RWSV_SAVE_FAB:
                .BLKB   4
000FC RWSV_CHAN:
                .BLKB   4
00100 RWSV_XOR_BCB:
                .BLKB   4
00104 RWSV_IN_SEQ:
                .BLKB   4
00108 RWSV_IN_SEQ_0:
                .BLKB   4
0010C RWSV_IN_XOR_SEQ:
                .BLKB   4
00110 RWSV_IN_XOR_RFA:
                .BLKB   6
00116 RWSV_LOOKAHEAD:
                .BLKB   1
00117 RWSV_XORSIZE:
                .BLKB   1
00118 RWSV_IN_GROUP_SIZE:
                .BLKB   4
0011C RWSV_IN_ERRORS:
                .BLKB   2
0011E RWSV_IN_XORUSE:
                .BLKB   2
00120 RWSV_IN_ORGERR:
                .BLKB   8
00128 RWSV_IN_VBN:
                .BLKB   4
0012C RWSV_IN_VBN_0:
                .BLKB   4
00130 RWSV_ALLOC:
                .BLKB   4
00134 RWSV_EOF:
                .BLKB   4
00138 RWSV_OUT_SEQ:
                .BLKB   4
0013C RWSV_OUT_VBN:
                .BLKB   4
00140 RWSV_OUT_BLOCK_COUNT:
                .BLKB   4
00144 RWSV_OUT_ERRORS:
                .BLKB   2
00146 RWSV_SEQ_ERRORS:
                .BLKB   2
00148 RWSV_OUT_GROUP_COUNT:
                .BLKB   1
00149 RWSV_PADDING:
```

```
                                .BLKB   3
0014C QUAL:     .BLKB   112
001BC COM_SSNAME:
                                .BLKB   8
001C4 COM_VALID_TYPES:
                                .BLKB   2
001C6 COM_FLAGS:
                                .BLKB   2
001C8 COM_PADDING:
                                .BLKB   1
001C9 COM_BUFF_COUNT:
                                .BLKB   1
001CA COM_I_SETCOUNT:
                                .BLKB   1
001CB COM_O_SETCOUNT:
                                .BLKB   1
001CC COM_I_STRUCNAME:
                                .BLKB   12
001D8 COM_O_STRUCNAME:
                                .BLKB   12
001E4 COM_O_BSRDATE:
                                .BLKB   8
001EC ALT_SSNAME:
                                .BLKB   32
0020C INPUT_FUNC:
                                .BLKB   1
0020D INPUT_RTYPE:
                                .BLKB   1
0020E OUTPUT_FUNC:
                                .BLKB   1
0020F FAST_STRUCLEV:
                                .BLKB   1
00210 INPUT_BEG:
                                .BLKB   0
00210 INPUT_CHAN:
                                .BLKB   4
00214 INPUT_FLAGS:
                                .BLKB   2
00216 INPUT_PADDING:
                                .BLKB   2
00218 INPUT_FAB:
                                .BLKB   4
0021C INPUT_NAM:
                                .BLKB   4
00220 INPUT_BCB:
                                .BLKB   4
00224 INPUT_QUAL:
                                .BLKB   4
00228 INPUT_BAD:
                                .BLKB   4
0022C INPUT_BLOCK:
                                .BLKB   4
00230 INPUT_MAXBLOCK:
                                .BLKB   4
00234 INPUT_MEDIA_ID:
                                .BLKB   4
00238 INPUT_NAMEDESC:
```

```
                                .BLKB   8
00240 INPUT_STATBLK:
                                .BLKB   8
00248 INPUT_HDR_BEG:
                                .BLKB   0
00248 INPUT_CREDATE:
                                .BLKB   8
00250 INPUT_REVDATE:
                                .BLKB   8
00258 INPUT_EXPDATE:
                                .BLKB   8
00260 INPUT_BAKDATE:
                                .BLKB   8
00268 INPUT_FILEOWNER:
                                .BLKB   4
0026C INPUT_FILECHAR:
                                .BLKB   4
00270 INPUT_RECATTR:
                                .BLKB   32
00290 INPUT_HDR_END:
                                .BLKB   0
00290 INPUT_END:
                                .BLKB   0
00290 INPUT_PROC_LIST:
                                .BLKB   4
00294 INPUT_PLACEMENT:
                                .BLKB   8
0029C INPUT_VBN_LIST:
                                .BLKB   8
002A4 INPUT_PLACE_LEN:
                                .BLKB   2
002A6 INPUT_PADDING_2:
                                .BLKB   2
002A8 OUTPUT_BEG:
                                .BLKB   0
002A8 OUTPUT_CHAN:
                                .BLKB   4
002AC OUTPUT_FLAGS:
                                .BLKB   2
002AE OUTPUT_PADDING:
                                .BLKB   2
002B0 OUTPUT_FAB:
                                .BLKB   4
002B4 OUTPUT_NAM:
                                .BLKB   4
002B8 OUTPUT_BCB:
                                .BLKB   4
002BC OUTPUT_QUAL:
                                .BLKB   4
002C0 OUTPUT_BAD:
                                .BLKB   4
002C4 OUTPUT_BLOCK:
                                .BLKB   4
002C8 OUTPUT_MAXBLOCK:
                                .BLKB   4
002CC OUTPUT_DEVGEOM:
                                .BLKB   8
```

```
002D4 OUTPUT_ATTBUF:
                .BLKB   144
00364 OUTPUT_END:
                .BLKB   0
00364 LIST_TOTFILES:
                .BLKB   4
00368 LIST_TOTSIZE:
                .BLKB   4
0036C VERIFY_FAB:
                .BLKB   4
00370 VERIFY_USE_COUNT:
                .BLKB   4
00374 VERIFY_QUAL:
                .BLKB   4
00378 COMPARE_BCB:
                .BLKB   4
0037C FAST_BUFFER:
                .BLKB   4
00380 FAST_BUFFER_SIZE:
                .BLKB   4
00384 FAST_RVN:
                .BLKB   1
00385 FAST_PADDING:
                .BLKB   1
00386 DIR_VERLIMIT:
                .BLKB   2
00388 FAST_VOL_BEG:
                .BLKB   0
00388 FAST_IMAP_SIZE:
                .BLKB   4
0038C FAST_IMAP:
                .BLKB   4
00390 FAST_HDR_OFFSET:
                .BLKB   4
00394 FAST_BOOT_LBN:
                .BLKB   4
00398 FAST_VOL_END:
                .BLKB   0
00398 JOUR_BUFFER:
                .BLKB   4
0039C JOUR_DIR:
                .BLKB   4
003A0 JOUR_HIBLK:
                .BLKB   4
003A4 JOUR_EFBLK:
                .BLKB   4
003A8 JOUR_INBLK:
                .BLKB   4
003AC JOUR_FFBYTE:
                .BLKB   2
003AE JOUR_INBYTE:
                .BLKB   2
003B0 JOUR_STRUCT_LEV:
                .BLKB   2
003B2 JOUR_COUNT:
                .BLKB   1
003B3 JOUR_REVERSE:
```

```
                                                        .BLKB   1
                                003B4 JOUR_EXSZ:
                                                        .BLKB   2
                                003B6 JOUR_PADDING:
                                                        .BLKB   2
                                003B8 CHKPT_HIGH_SP:
                                                        .BLKB   4
                                003BC CHKPT_LOW_SP:
                                                        .BLKB   4
                                003C0 CHKPT_STACK:
                                                        .BLKB   4
                                003C4 CHKPT_VARS:
                                                        .BLKB   4
                                003C8 CHKPT_STATUS:
                                                        .BLKB   4
                                003CC DIR_BEG:.BLKB     0
                                003CC DIR_CHAN:
                                                        .BLKB   4
                                003D0 DIR_NAM:.BLKB     4
                                003D4 DIR_DEV_DESC:
                                                        .BLKB   4
                                003D8 DIR_SEL_DIR:
                                                        .BLKB   8
                                003E0 DIR_SEL_NTV:
                                                        .BLKB   8
                                003E8 DIR_STRUCLEV:
                                                        .BLKB   1
                                003E9 DIR_LEVELS:
                                                        .BLKB   1
                                003EA DIR_FLAGS:
                                                        .BLKB   1
                                003EB DIR_STATUS:
                                                        .BLKB   1
                                003EC DIR_STRING:
                                                        .BLKB   320
                                0052C DIR_STACK:
                                                        .BLKB   612
                                00790 DIR_SP:  .BLKB    4
                                00794 DIR_SEL_LATEST:
                                                        .BLKB   4
                                00798 DIR_END:.BLKB     0
                                00798 DIR_SCANLIMIT:
                                                        .BLKB   36
                                007BC INPUT_MTL:
                                                        .BLKB   4
                                007C0 OUTPUT_MTL:
                                                        .BLKB   4
                                007C4 CURRENT_MTL:
                                                        .BLKB   4
                                007C8 CURRENT_VCB:
                                                        .BLKB   4
                                007CC CURRENT_WCB:
                                                        .BLKB   4
                                007D0 ACL_FIB_DESCR:
                                                        .BLKB   8
                                007D8 ACL_FIB:.BLKB     64
                                00818 ACL_LENGTH:
```

```
                                                .BLKB   4
                                        0081C ACL_BUFFER:
                                                .BLKB   4
                                        00820 CRYP_IN_CONTEXT:
                                                .BLKB   4
                                        00824 CRYP_OU_CONTEXT:
                                                .BLKB   4
                                        00828 CRYP_DA_CONTEXT:
                                                .BLKB   4
                                        0082C CRYP_DATA_ENCIV:
                                                .BLKB   8
                                        00834 CRYP_DATA_CODE:
                                                .BLKB   4
                                        00838 CRYP_DATA_KEY:
                                                .BLKB   8
                                        00840 CRYP_DATA_IV:
                                                .BLKB   8
                                        00848 CRYP_DATA_CKSM:
                                                .BLKB   4

                                                .PSECT  CODE,NOWRT,2

                                     0A 00000 P.AAB:    .ASCII  <10>                                    ;
                                   20 24 00001          .ASCII  \$ \                                    ;
                                         00003          .BLKB   1
                                00000003 00004 P.AAA:   .LONG   3                                       ;
                                00000000' 00008         .ADDRESS P.AAB                                  ;

                                                .EXTRN  BOO$ACTIMAGE, CLI$DCL_PARSE
                                                .EXTRN  LIB$GET_COMMAND
                                                .EXTRN  CON$PUTCHAR, CON$OWNCTY
                                                .EXTRN  EXE$GL_FLAGS, VERSION_STRING
                                                .EXTRN  BACKUPCMD, BACKUP$_IDENT
                                                .EXTRN  EXE$V_INIT, SYS$CMKRNL

                                         0000 00000             .ENTRY  STA_INIT, Save nothing          ; 1212
                                   5E    FF74 CE   9E 00002     MOVAB   -140(SP), SP                    ;
                                     0C  BD   0000V CF   9E 00007     MOVAB   STA_HANDLER, @12(FP)      ; 1253
                              0D 00000000G 00     00G E0 0000D     BBS     S^EXE$V_INIT, EXE$GL_FLAGS, 1$  ; 1259
                                          7E   D4 00015     CLRL    -(SP)                           ; 1261
                                   0000V  CF   9F 00017     PUSHAB  STA_BUG_INSTALL                 ;
                                00000000G 00     02 FB 0001B     CALLS   #2, SYS$CMKRNL                  ;
                                          7E   D4 00022 1$:  CLRL    -(SP)                           ; 1267
                                00000000G 00     9F 00024     PUSHAB  VERSION_STRING                  ;
                                          04     DD 0002A     PUSHL   #4                              ;
                                          03     DD 0002C     PUSHL   #3                              ;
                                00000000G 8F     DD 0002E     PUSHL   #BACKUP$_IDENT                  ;
                                00000000G 00     05 FB 00034     CALLS   #5, LIB$SIGNAL                  ;
                                00000000' EF     02 88 0003B     BISB2   #2, COM_FLAGS                   ; 1272
                                   6E 010E0084   8F   D0 00042 2$:  MOVL    #17694852, DESC         ; 1275
                                         04 AE      08 AE 9E 00049     MOVAB   BUFFER, DESC+4          ; 1278
                                          5E   DD 0004E     PUSHL   SP                              ; 1279
                                               A5   AF 9F 00050     PUSHAB  P.AAA                   ;
                                               08   AE 9F 00053     PUSHAB  DESC                    ;
                                00000000G 00        03 FB 00056     CALLS   #3, LIB$GET_COMMAND             ;
                                                    6E B5 0005D     TSTW    DESC                    ; 1283
                                                    E1 13 0005F     BEQL    2$                      ;
```

```
                        00000000G  00   9F 00061          PUSHAB  BACKUPCMD                          ; 1285
                               04  AE   9F 00067          PUSHAB  DESC                               ;
               00000000G  00            02 FB 0006A       CALLS   #2, CLI$DCL_PARSE                  ;
                          CE            50 E9 00071       BLBC    R0, 2$                             ;
                                           04 00074       RET                                        ; 1287

; Routine Size:  117 bytes,    Routine Base:  CODE + 000C
```

```
;  186     1288  1 %SBTTL 'STA_BUGCHECK - system bugcheck overlay'
;  187     1289  1 OWN STA_BUGCHECK_BEGIN: PSECT(CODE) VECTOR[0];
;  188     1290  1 FORWARD STA_BUGCHECK_MESSAGE: VECTOR[14,BYTE];
;  189     1291  1 ROUTINE STA_BUGCHECK: JSB NOVALUE=
;  190     1292  1
;  191     1293  1 !++
;  192     1294  1 !
;  193     1295  1 ! FUNCTIONAL DESCRIPTION:
;  194     1296  1 !       This routine is copied over EXE$BUG_CHECK by the initialization logic
;  195     1297  1 !       in order to trap bugchecks that might occur during execution of
;  196     1298  1 !       standalone BACKUP.
;  197     1299  1 !
;  198     1300  1 ! INPUT PARAMETERS:
;  199     1301  1 !       NONE
;  200     1302  1 !
;  201     1303  1 ! IMPLICIT INPUTS:
;  202     1304  1 !       NONE
;  203     1305  1 !
;  204     1306  1 ! OUTPUT PARAMETERS:
;  205     1307  1 !       NONE
;  206     1308  1 !
;  207     1309  1 ! IMPLICIT OUTPUTS:
;  208     1310  1 !       NONE
;  209     1311  1 !
;  210     1312  1 ! ROUTINE VALUE:
;  211     1313  1 !       NONE
;  212     1314  1 !
;  213     1315  1 ! SIDE EFFECTS:
;  214     1316  1 !       NONE
;  215     1317  1 !
;  216     1318  1 !--
;  217     1319  1
;  218     1320  2 BEGIN
;  219     1321  2 BUILTIN
;  220     1322  2         HALT,
;  221     1323  2         MFPR,
;  222     1324  2         MTPR;
;  223     1325  2 LOCAL
;  224     1326  2         P:              REF VECTOR[,BYTE],
;  225     1327  2         S;
;  226     1328  2
;  227     1329  2
;  228     1330  2 MTPR(%REF(IPL$_POWER), PR$_IPL);
;  229     1331  2 CON$OWNCTY();
;  230     1332  2 P = STA_BUGCHECK_MESSAGE;
;  231     1333  2 DECR L FROM 14 TO 1 DO
;  232     1334  3     BEGIN
;  233     1335  3     S = .P[0];  P = .P + 1;  CON$PUTCHAR(.S);
;  234     1336  2     END;
;  235     1337  2 HALT();
;  236     1338  1 END;


                                                00081          .BLKB   3
                                                00084 STA_BUGCHECK_BEGIN:
                                                               .BLKB   0
```

```
                                 12            1F  DA 00000 STA_BUGCHECK:
                                                            MTPR    #31, #18                                ; 1330
                                    00000000G   00  16 00003            JSB     CON$OWNCTY                              ; 1331
                                 52     0000V   CF  9E 00009            MOVAB   STA_BUGCHECK_MESSAGE, P         ; 1332
                                 51             0E  D0 0000E            MOVL    #14, L                          ; 1333
                                 50             82  9A 00011 1$:        MOVZBL  (P)+, S                         ; 1335
                                    00000000G   00  16 00014            JSB     CON$PUTCHAR                     ;
                                 F4             51  F5 0001A            SOBGTR  L, 1$                           ; 1333
                                                    00 0001D            HALT                                    ; 1337
                                                    05 0001E            RSB                                     ; 1338

; Routine Size:  31 bytes,    Routine Base:  CODE + 0084

;  237          1339  1 OWN STA_BUGCHECK_MESSAGE: PSECT(CODE) VECTOR[14,BYTE]
;  238          1340  1 INITIAL(BYTE
;  239          1341  1     (%CHAR(%O'015', %O'012', 0, 0), 'Bugcheck', %CHAR(%O'015', %O'012')));
;  240          1342  1 OWN STA_BUGCHECK_END: PSECT(CODE) VECTOR[0];
```

```
;  242          1343  1 %SBTTL 'STA_BUG_INSTALL - install bugcheck overlay'
;  243          1344  1 ROUTINE STA_BUG_INSTALL=
;  244          1345  1
;  245          1346  1 !++
;  246          1347  1 !
;  247          1348  1 ! FUNCTIONAL DESCRIPTION:
;  248          1349  1 !       This routine is called in kernel mode to install the bugcheck routine
;  249          1350  1 !       over EXE$BUG_CHECK.
;  250          1351  1 !
;  251          1352  1 ! INPUT PARAMETERS:
;  252          1353  1 !       NONE
;  253          1354  1 !
;  254          1355  1 ! IMPLICIT INPUTS:
;  255          1356  1 !       Code between STA_BUGCHECK_BEGIN and STA_BUGCHECK_END.
;  256          1357  1 !
;  257          1358  1 ! OUTPUT PARAMETERS:
;  258          1359  1 !       NONE
;  259          1360  1 !
;  260          1361  1 ! IMPLICIT OUTPUTS:
;  261          1362  1 !       Code between STA_BUGCHECK_BEGIN and STA_BUGCHECK_END moved to
;  262          1363  1 !       EXE$BUG_CHECK.
;  263          1364  1 !
;  264          1365  1 ! ROUTINE VALUE:
;  265          1366  1 !       SS$_NORMAL
;  266          1367  1 !
;  267          1368  1 ! SIDE EFFECTS:
;  268          1369  1 !       NONE
;  269          1370  1 !
;  270          1371  1 !--
;  271          1372  1
;  272          1373  2 BEGIN
;  273          1374  2 LINKAGE
;  274          1375  2         INI=                    JSB:    PRESERVE(0,1,2,3,4,5,6,7,8,9,10,11);
;  275          1376  2 EXTERNAL ROUTINE
;  276          1377  2         INI$WRITABLE:           INI NOVALUE ADDRESSING_MODE(GENERAL),
;  277          1378  2         INI$RDONLY:             INI NOVALUE ADDRESSING_MODE(GENERAL);
;  278          1379  2 EXTERNAL LITERAL
;  279          1380  2         EXE$V_BUGDUMP   : UNSIGNED (6);
;  280          1381  2 EXTERNAL
;  281          1382  2         EXE$GL_FLAGS    : BITVECTOR ADDRESSING_MODE (GENERAL),
;  282          1383  2         EXE$BUG_CHECK:          ADDRESSING_MODE(GENERAL);
;  283          1384  2
;  284          1385  2
;  285          1386  2 IF NOT .EXE$GL_FLAGS[EXE$V_BUGDUMP]
;  286          1387  2 THEN
;  287          1388  3     BEGIN
;  288          1389  3     INI$WRITABLE();             ! Make system writable
;  289          1390  3     CH$MOVE(STA_BUGCHECK_END-STA_BUGCHECK_BEGIN, STA_BUGCHECK_BEGIN, EXE$BUG_CHECK);
;  290          1391  3     INI$RDONLY();               ! Make system read-only
;  291          1392  2     END;
;  292          1393  2
;  293          1394  2 SS$_NORMAL
;  294          1395  1 END;

                                        000A3          .BLKB   1
```

```
                                                00  00  0A  0D  000A4 STA_BUGCHECK_MESSAGE:
                                                                        .ASCII  <13><10><0><0>                          ;
                                        6B  63  65  68  63  67  75  42  000A8           .ASCII  \Bugcheck\                              ;
                                                        0A  0D  000B0           .ASCII  <13><10>                                ;
                                                                000B2           .BLKB   2
                                                                000B4 STA_BUGCHECK_END:
                                                                        .BLKB   0

                                                                        .EXTRN  INI$WRITABLE, INI$RDONLY
                                                                        .EXTRN  EXE$V_BUGDUMP, EXE$BUG_CHECK

                                                            0FFC 00000 STA_BUG_INSTALL:
                                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 1344
                                    15 00000000G    00              00G E0 00002        BBS     S^EXE$V_BUGDUMP, EXE$GL_FLAGS, 1$       ; 1386
                                                        00000000G   00  16 0000A        JSB     INI$WRITABLE                            ; 1389
                        00000000G   00      BC  AF                  30  28 00010       MOVC3   #48, STA_BUGCHECK_BEGIN, EXE$BUG_CHECK  ; 1390
                                                        00000000G   00  16 00019        JSB     INI$RDONLY                              ; 1391
                                                50                  01  D0 0001F 1$:    MOVL    #1, R0                                  ; 1395
                                                                    04  00022        RET                                             ;

; Routine Size:  35 bytes,    Routine Base:  CODE + 00B4
```

```
;   296          1396  1 %SBTTL 'STA_HANDLER - top level condition handler'
;   297          1397  1 ROUTINE STA_HANDLER(SIGNAL,MECHANISM)=
;   298          1398  1
;   299          1399  1 !++
;   300          1400  1 !
;   301          1401  1 ! FUNCTIONAL DESCRIPTION:
;   302          1402  1 !       This routine is established as a stack condition handler in the main
;   303          1403  1 !       routine for the standalone version.  It calls $PUTMSG to output a
;   304          1404  1 !       signalled message.  Then, if the message is fatal, it calls STA_RESTART
;   305          1405  1 !       to start the image over (or exit).
;   306          1406  1 !
;   307          1407  1 ! INPUT PARAMETERS:
;   308          1408  1 !       SIGNAL          - Standard VMS condition handler
;   309          1409  1 !       MECHANISM       -     parameters
;   310          1410  1 !
;   311          1411  1 ! IMPLICIT INPUTS:
;   312          1412  1 !       NONE
;   313          1413  1 !
;   314          1414  1 ! OUTPUT PARAMETERS:
;   315          1415  1 !       NONE
;   316          1416  1 !
;   317          1417  1 ! IMPLICIT OUTPUTS:
;   318          1418  1 !       NONE
;   319          1419  1 !
;   320          1420  1 ! ROUTINE VALUE:
;   321          1421  1 !       SS$_CONTINUE
;   322          1422  1 !
;   323          1423  1 ! SIDE EFFECTS:
;   324          1424  1 !       If the message is of fatal severity, image is re-activated (or exits).
;   325          1425  1 !
;   326          1426  1 !--
;   327          1427  1
;   328          1428  2 BEGIN
;   329          1429  2 MAP
;   330          1430  2         SIGNAL:         REF BBLOCK,     ! Signal parameters
;   331          1431  2         MECHANISM:      REF BBLOCK;     ! Mechanism parameters
;   332          1432  2
;   333          1433  2
;   334          1434  2 IF .SIGNAL[CHF$L_SIG_NAME] NEQ SS$_UNWIND
;   335          1435  2 THEN
;   336          1436  3     BEGIN
;   337          1437  3
;   338          1438  3     ! Call $PUTMSG to issue the message, after stripping the PC and PSL from
;   339          1439  3     ! the signal arguments.
;   340          1440  3     !
;   341          1441  3     SIGNAL[CHF$L_SIG_ARGS] = .SIGNAL[CHF$L_SIG_ARGS] - 2;
;   342          1442  3     $PUTMSG(MSGVEC=.SIGNAL, ACTRTN=PUTMSG_ACTRTN);
;   343          1443  3
;   344          1444  3
;   345          1445  3     ! If the message was fatal, restart the image (or exit).
;   346          1446  3     !
;   347          1447  3     IF .BBLOCK[SIGNAL[CHF$L_SIG_NAME], STS$V_SEVERITY] EQL STS$K_SEVERE
;   348          1448  3     THEN
;   349          1449  3         STA_RESTART();
;   350          1450  2     END;
;   351          1451  2
;   352          1452  2
```

```
;  353          1453  2 SS$_CONTINUE
;  354          1454  1 END;


                                                                        .EXTRN   SYS$PUTMSG

                                               0004 00000 STA_HANDLER:
                                                                        .WORD    Save R2                                  ; 1397
                            52          04   AC   D0 00002              MOVL     SIGNAL, R2                               ; 1434
                  00000920  8F          04   A2   D1 00006              CMPL     4(R2), #2336                             ;
                                             1F   13 0000E              BEQL     1$                                       ;
                            62               02   C2 00010              SUBL2    #2, (R2)                                 ; 1441
                                             7E   7C 00013              CLRQ     -(SP)                                    ; 1442
                                    0000V    CF   9F 00015              PUSHAB   PUTMSG_ACTRTN                            ;
                                             52   DD 00019              PUSHL    R2                                       ;
                  00000000G 00               04   FB 0001B              CALLS    #4, SYS$PUTMSG                           ;
        04        04   A2   03               00   ED 00022              CMPZV    #0, #3, 4(R2), #4                        ; 1447
                                             05   12 00028              BNEQ     1$                                       ;
                      0000V CF               00   FB 0002A              CALLS    #0, STA_RESTART                          ; 1449
                            50               01   D0 0002F 1$:          MOVL     #1, R0                                   ; 1454
                                                  04 00032              RET                                               ;

; Routine Size:  51 bytes,    Routine Base:  CODE + 00D7
```

```
;  356          1455  1 %SBTTL 'PUTMSG_ACTRTN - $PUTMSG action routine'
;  357          1456  1 GLOBAL ROUTINE PUTMSG_ACTRTN(DESC)=
;  358          1457  1
;  359          1458  1 !++
;  360          1459  1 !
;  361          1460  1 ! FUNCTIONAL DESCRIPTION:
;  362          1461  1 !       This routine is the $PUTMSG action routine for the general
;  363          1462  1 !       handler.  It uses the stand-alone LIB$PUT_OUTPUT to write each
;  364          1463  1 !       message line on the terminal.
;  365          1464  1 !
;  366          1465  1 ! INPUT PARAMETERS:
;  367          1466  1 !       DESC            - Descriptor for message line.
;  368          1467  1 !
;  369          1468  1 ! IMPLICIT INPUTS:
;  370          1469  1 !       NONE
;  371          1470  1 !
;  372          1471  1 ! OUTPUT PARAMETERS:
;  373          1472  1 !       NONE
;  374          1473  1 !
;  375          1474  1 ! IMPLICIT OUTPUTS:
;  376          1475  1 !       NONE
;  377          1476  1 !
;  378          1477  1 ! ROUTINE VALUE:
;  379          1478  1 !       False, to prevent $PUTMSG from writing the line.
;  380          1479  1 !
;  381          1480  1 ! SIDE EFFECTS:
;  382          1481  1 !       NONE
;  383          1482  1 !
;  384          1483  1 !--
;  385          1484  1
;  386          1485  2 BEGIN
;  387          1486  2 EXTERNAL ROUTINE
;  388          1487  2         LIB$PUT_OUTPUT;
;  389          1488  2
;  390          1489  2
;  391          1490  2 LIB$PUT_OUTPUT(.DESC);
;  392          1491  2 FALSE
;  393          1492  1 END;


                                                        .EXTRN  LIB$PUT_OUTPUT

                                        0000 00000      .ENTRY  PUTMSG_ACTRTN, Save nothing                     ; 1456
                                    04  AC   DD 00002  PUSHL   DESC                                           ; 1490
                        00000000G   00      01   FB 00005  CALLS   #1, LIB$PUT_OUTPUT                        ;
                                            50   D4 0000C  CLRL    R0                                         ; 1492
                                                 04 0000E  RET                                                 ;

; Routine Size:  15 bytes,    Routine Base:  CODE + 010A
```

```
1 %SBTTL 'STA_RESTART - stand-alone image restart'
1 GLOBAL ROUTINE STA_RESTART: NOVALUE=
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is called when image execution should terminate.  If
1 !       running online, it exits.  If running standalone, it restarts by
1 !       calling BOO$ACTIMAGE to re-activate the image.
1 !
1 ! INPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       EXE$GL_FLAGS[EXE$V_INIT] - True if running online.
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       Image is re-activated (or exits).  Thus, control should not return.
1 !
1 !--
1
2 BEGIN
2
2 ! If running standalone, re-activate the image.
2 !
2 IF NOT .EXE$GL_FLAGS[EXE$V_INIT]
2 THEN
2     BOO$ACTIMAGE($DESCRIPTOR('STANDALON.EXE'));
2
2
2 ! Re-activate failed or not executed.
2 !
2 $EXIT();
1 END;


 45 58 45 2E 4E 4F 4C 41 44 4E 41 54 53  00119 P.AAD:  .ASCII  \STANDALON.EXE\             ;
                                         00126         .BLKB   2
                              0000000D   00128 P.AAC:  .LONG   13                          ;
                              00000000'  0012C         .ADDRESS P.AAD                      ;

                                                       .EXTRN  SYS$EXIT

                                   0000  00000         .ENTRY  STA_RESTART, Save nothing   ; 1494
             0A 00000000G  00     00G E0 00002         BBS     S^EXE$V_INIT, EXE$GL_FLAGS, 1$ ; 1527
                              EB   AF 9F 0000A         PUSHAB  P.AAC                       ; 1529
                00000000G  00      01 FB 0000D         CALLS   #1, BOO$ACTIMAGE            ;
```

```
                                    01  DD 00014 1$:     PUSHL   #1                          ; 1534
                  00000000G  00     01  FB 00016         CALLS   #1, SYS$EXIT                ;
                                        04 0001D         RET                                 ; 1535

; Routine Size:  30 bytes,    Routine Base:  CODE + 0130
```

```
;  439          1536  1 END
;  440          1537  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  COMMON                         2124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  OVR,NOPIC,ALIGN(2)
;  CODE                            334  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1          18619        23         0        1000        00:02.2


;                                       COMMAND QUALIFIERS

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

; Size:         279 code + 2179 data bytes
; Run Time:         00:18.6
; Elapsed Time:     00:57.1
; Lines/CPU Min:     4971
; Lexemes/CPU-Min: 42401
; Memory Used:  217 pages
; Compilation Complete
```

STABACCOP
LIS

STAINIT
LIS

TAPEUTIL
LIS

STAINVOL
LIS

UTILITY
LIS

STATOSYS
LIS

STAPARSE
LIS