E 14

```
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCCC  HHH         HHH
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCCC  HHH         HHH
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCCC  HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPP         PPP  AAA         AAA        TTT        CCC              HHH         HHH
PPPPPPPPPPPP     AAA         AAA        TTT        CCC              HHHHHHHHHHHHHHH
PPPPPPPPPPPP     AAA         AAA        TTT        CCC              HHHHHHHHHHHHHHH
PPPPPPPPPPPP     AAA         AAA        TTT        CCC              HHHHHHHHHHHHHHH
PPP              AAAAAAAAAAAAAAA        TTT        CCC              HHH         HHH
PPP              AAAAAAAAAAAAAAA        TTT        CCC              HHH         HHH
PPP              AAAAAAAAAAAAAAA        TTT        CCC              HHH         HHH
PPP              AAA         AAA        TTT        CCC              HHH         HHH
PPP              AAA         AAA        TTT        CCC              HHH         HHH
PPP              AAA         AAA        TTT        CCC              HHH         HHH
PPP              AAA         AAA        TTT           CCCCCCCCCCCC  HHH         HHH
PPP              AAA         AAA        TTT           CCCCCCCCCCCC  HHH         HHH
PPP              AAA         AAA        TTT           CCCCCCCCCCCC  HHH         HHH
```

FILEID**PATSPA

```
PPPPPPPP      AAAAAA    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      AAAAAA
PPPPPPPP      AAAAAA    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      AAAAAA
PP      PP  AA      AA      TT      SS          PP      PP  AA      AA
PP      PP  AA      AA      TT      SS          PP      PP  AA      AA
PP      PP  AA      AA      TT      SS          PP      PP  AA      AA
PP      PP  AA      AA      TT      SS          PP      PP  AA      AA
PPPPPPPP    AA      AA      TT        SSSSSS    PPPPPPPP    AA      AA
PPPPPPPP    AA      AA      TT        SSSSSS    PPPPPPPP    AA      AA
PP          AAAAAAAAAA      TT              SS  PP          AAAAAAAAAA
PP          AAAAAAAAAA      TT              SS  PP          AAAAAAAAAA
PP          AA      AA      TT              SS  PP          AA      AA      ....
PP          AA      AA      TT              SS  PP          AA      AA      ....
PP          AA      AA      TT      SSSSSSSS    PP          AA      AA      ....
PP          AA      AA      TT      SSSSSSSS    PP          AA      AA      ....

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
```

PAT
V04

```
MODULE PATSPA (%IF %VARIANT EQL 1
                %THEN
                        ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE,
                                         NONEXTERNAL = LONG_RELATIVE),
                %FI
                IDENT = 'V04-000'
                ) =
BEGIN
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!

!++
! FACILITY:     PATCH
!
! ABSTRACT:     THIS ROUTINE HANDLES FREE PATCH AREA, ALIGNMENT, ALLOCATION, AND EXPANSION.
!
! ENVIRONMENT:  VAX/VMS
!
! AUTHOR: K.D. MORSE     , CREATION DATE: 17-NOV-77
!
! MODIFIED BY:
!
!       V03-003 MTR0025         Mike Rhodes                     11-Aug-1983
!               Modify routine PAT$EXP_AREA to signal an ERROR (severity)
!               message when an expansion request is made while patching
!               a file in ABSOLUTE mode.  This will cause the current
!               command to be aborted and the user is returned back to the
!               PATCH command prompt.  Files may NOT be expanded in absolute
!               mode, as could result from a command like:
!                       PATCH>  REPLACE/INST 20='movl r0,r1'
!                       NEW>    'movl r0,r1'
!                       NEW>    'bneq 200'
!                       NEW>    EXIT
!
```

```
1 !       V03-002 MTR0016          Mike Rhodes              03-Nov-1982
1 !               Modify PAT$BUILD_ISE to accept one additional argument which
1 !               is the address to be modified.  This address is used for INSERT
1 !               and REPLACE commands when patching protected shareable images.
1 !               The attributes of the image section which contains the address
1 !               being modified will be propagated to the newly created default
1 !               patch area.
1 !
1 !       V03-001 MTR0007          Mike Rhodes              14-Jun-1982
1 !               Use shared system messages.  Affected modules include:
1 !               DYNMEM.B32, PATBAS.B32, PATCMD.B32, PATIHD.B32, PATINT.B32,
1 !               PATIO.B32, PATMAI.B32, PATMSG.MSG, PATWRT.B32, and PATSPA.B32.
1 !
1 !               The shared messages are defined by DYNMEM.B32's invocation of
1 !               SHRMSG.REQ and we simply link against these symbols.  They are
1 !               declared as external literals below.
1 !
1 !       V03-000 MTR0001          Mike Rhodes              15-Mar-1982
1 !               Modify routine PAT$EXP_AREA to allow PIC SHR images to be
1 !               patched using default patch area which may be expanded as
1 !               needed.  Also, removed the old 50% growth area logic which
1 !               has been made obsolete by the above change.
1 !
1 !       V02-008 MTR0001          Mike Rhodes              15-Sep-1981
1 !               Modify routine PAT$BUILD_ISE.  The location algorithm
1 !               for placing the PATCH ISE/ISD pair in the ISE list is
1 !               as follows:
1 !                       The PATCH ISE/ISD pair are located in the ISE list
1 !                       FOLLOWING the last "Normal" ISD and PRECEDING the
1 !                       first Non-Based Global or Stack ISDs.
1 !
1 !               Included in the modification is the definition of two new
1 !               variables,  PREV_ISE_PTR - Pointer to Previous ISE,  and
1 !                           TEMP         - Holds the FLINK from the previous
1 !                                          ISE till its put into the new ISE.
1 !
1 !       V02-007 PCG0001          Peter George             02-FEB-1981
1 !               Add require statement for LIB$:PATDEF.REQ
1 !
1 !       V0206   CNH0038          Chris Hume               4-Oct-1980     16:00
1 !               Last Cluster will now remain set when new Patch Area is added.
1 !               Patch Area will be allocated at a distance one half the size of
1 !               the Last Cluster (beyond its end).
1 !
1 !       V0105   CNH0023          Chris Hume               16-Nov-1979    14:00
1 !               Turn off ISD$V_LASTCLU for all ISD's when PATCH Area is added
1 !               to an image.  Also unrecognized languages will now be processed
1 !               as though they were MACRO.  (PATBLD.B32 V0117, PATMAI.B32 V0228)
1 !
1 !       V0104   CNH0015          Chris Hume               27-Sep-1979    11:30
1 !               Changed GBLWARN message from a warning to an informational.
1 !               Added section name to the signal.  Added EXPSHRPAT error.
1 !               (PATMAI.B32 V0225, PATMSG.MDL V0203, PATARI.B32 V0112)
1 !
1 ! MODIFICATIONS:
1 !
1 !  NO   DATE            PROGRAMMER      PURPOSE
```

```
;  115        0115  1 !  --    ----             ----------        -------
;  116        0116  1 !
;  117        0117  1 !  01    07-MAR-78        K.D. MORSE        ADD ROUTINES PAT$ADD_PAL.
;  118        0118  1 !  02    25-APR-78        K.D. MORSE        CONVERT TO NATIVE COMPILER.
;  119        0119  1 !  03    13-JUN-78        K.D. MORSE        ADD FAO COUNTS TO SIGNALS.
;  120        0120  1 !
;  121        0121  1 !--
```

```
0122  1 !
0123  1 ! TABLE OF CONTENTS:
0124  1 !
0125  1
0126  1 FORWARD ROUTINE
0127  1         PAT$ALIGN_CMD : NOVALUE,                        ! Executes align command
0128  1         PAT$BUILD_ISE : NOVALUE,                        ! Builds an image section descriptor
0129  1         PAT$EXP_AREA : NOVALUE,                         ! Expands patch area
0130  1         PAT$ADD_PAL : NOVALUE;                          ! Adds entry to PAL
0131  1
0132  1 !
0133  1 ! INCLUDE FILES:
0134  1 !
0135  1 LIBRARY 'SYS$LIBRARY:LIB.L32';                         ! System structure definitions
0136  1 REQUIRE 'SRC$:PATPCT.REQ';                             ! Defines PSECTs
0176  1 REQUIRE 'SRC$:PATGEN.REQ';                             ! Defines context bits
0398  1 REQUIRE 'SRC$:VXSMAC.REQ';                             ! Defines common macros
0463  1 REQUIRE 'SRC$:PREFIX.REQ';                             ! Defines structure macros
0651  1 REQUIRE 'SRC$:PATPRE.REQ';                             ! Defines PATCH structures
0814  1 REQUIRE 'LIB$:PATDEF.REQ';                    ! Defines literals
0868  1 REQUIRE 'LIB$:PATMSG.REQ';                             ! Defines error message codes
1042  1 REQUIRE 'SRC$:BSTRUC.REQ';                             ! Defines basic structures
1118  1 REQUIRE 'SRC$:LISTEL.REQ';                             ! Defines list structures
1160  1 REQUIRE 'SRC$:DLLNAM.REQ';                             ! Defines symbol table entry offsets
1218  1 REQUIRE 'SRC$:SYSSER.REQ';                             ! Defines FAO output macros
```

```
;  R1250  1      SWITCHES LIST (SOURCE);
;  R1251  1
;  R1252  1      EXTERNAL ROUTINE
;  R1253  1              PAT$fao_out;            ! formats a line and outputs to the terminal
;  R1254  1
```

```
1 !
1 ! MACROS:
1 !
1
1 !
1 ! EQUATED SYMBOLS:
1 !
1
1 !
1 ! OWN STORAGE:
1 !
1 OWN
1         PAT_AREA_NAME : VECTOR[4,BYTE] INITIAL(%ASCIC 'PAA'),    ! Next patch area name
1         PA_NAME_DSC : VECTOR[2,LONG] INITIAL(A_LONGWORD-A_BYTE, CH$PTR(PAT_AREA_NAME, 1)); ! String descript
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
1 EXTERNAL
1         PAT$GL_PAL_LHD : REF BLOCK[,BYTE],                       ! Patch area listhead
1         PAT$GL_ERRCODE,                                          ! Error code
1         PAT$GL_CONTEXT : BITVECTOR,                              ! Context bits
1         PAT$GL_FLAGS : BITVECTOR [32],                           ! CLI flags.
1         PAT$GL_IMGHDR : REF BLOCK[,BYTE],                        ! Pointer to image header
1         PAT$GL_PATAREA : REF BLOCK[,BYTE],                       ! Free patch area descriptor pointer
1         PAT$GL_IHPPTR : REF BLOCK[,BYTE],                        ! Pointer to patch area of image header
1         PAT$GL_ISELHD,                                           ! ISE List Head
1         PAT$GL_ISETAIL : REF BLOCK[,BYTE],                       ! Pointer to tail of ISE table
1         PAT$GL_NEWVPNMX,                                         ! Max VPN of image sections in new image
1         PAT$GL_NEWVBNMX,                                         ! Max VBN of image sections in new image
1         PAT$GL_IMGBLKS,                                          ! Number of blocks in new image
1         PAT$GL_ISVADDR : VECTOR[,LONG],                          ! Addresses of last image section mapped
1         PAT$GL_HEAD_LST,                                         ! Head of command argument list
1         PAT$GL_SYMTBPTR,                                         ! Pointer to current default symbol table
1         PAT$GL_SYMHEAD;                                          ! Pointer to listhead entry for user-defined
1
1 EXTERNAL ROUTINE
1         PAT$ALLOBLK : NOVALUE,                                   ! Allocates free storage
1         PAT$CREMAP : NOVALUE,                                    ! Creates and maps image sections
1         PAT$DEFINE_SYM : NOVALUE,                                ! Defines a symbol
1         PAT$FIND_SYM,                                            ! Find symbol definition
1         PAT$FREEZ,                                               ! Allocates and zeros free storage
1         PAT$MAP_ADDR : NOVALUE;                                  ! Maps an image address
1
1 EXTERNAL LITERAL
1 !
1 ! Define shared message references.  (resolved @ link time)
1 !
1         PAT$_CLOSEIN,                                            ! Error closing input file.
1         PAT$_CLOSEOUT,                                           ! Error closing output file.
1         PAT$_OPENIN,                                             ! Error opening input file.
1         PAT$_OPENOUT,                                            ! Error opening output file.
1         PAT$_READERR,                                            ! Error reading from file.
1         PAT$_SYSERROR,                                           ! System Service error.
1         PAT$_WRITEERR;                                           ! Error writing to file.
1
```

```
1 GLOBAL ROUTINE PAT$ALIGN_CMD  : NOVALUE =                              ! Performs align commands
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine aligns a free patch area to the requested boundary,
1 !       word, longword, quadword, or page.  The patch area bytes between the
1 !       old address and the rounded address are lost for patching purposes.
1 !       The symbol name provided in the command is entered into the symbol list
1 !       with a value of the patch area address.  If the free patch area is not
1 !       large enough to be rounded to the appropriate boundary, an error is
1 !       SIGNALed and the alignment does not take place.  The free area
1 !       descriptor remains unchanged.
1 !
1 !       If the symbol name was previously defined, a message is produced and
1 !       the name is redefined to the new patch area address.
1 !
1 !       Aligning the patch area to a byte boundary will merely cause the
1 !       symbol to be defined as the next free byte of patch area.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       none
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       The symbol name descriptor is set up by the parser.
1 !       The context bits have already been set up for the command.
1 !       The user-defined symbol table has been initialized as has the
1 !       free memory handler.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !
1 !       none
1 !
1 ! COMPLETION CODES:
1 !
1 !       none
1 !
1 ! SIDE EFFECTS:
1 !
1 !       The default patch area is aligned to the appropriate boundary.
1 !       If there is not enough patch area to align, a new patch area is
1 !       created.
1 !
1 !--
1
2 BEGIN
2
2 LITERAL
2       ONE_BLOCK = 1;                                          ! Number of blocks to expand patch area by
2
2 LOCAL
```

```
2               TEMP_SYMTB,                                             ! Temporary symbol table pointer
2               ALIGN_FACTOR,                                           ! Alignment boundary
2               DESC_PTR : REF BLOCK[,BYTE],                            ! String descriptor pointer
2               SYM_ENTRY_PTR,                                          ! Pointer to symbol entry
2               PATCH_AREA_ADR,                                         ! Address of aligned patch area
2               PATCH_AREA_SIZ;                                         ! Size of aligned patch area
2
2       !++
2       ! Output current patch area statistics before alignment.
2       !--
2       $FAO_TT_OUT('old patch area size:      !XL', .PAT$GL_PATAREA[DSC$W_LENGTH]);
2       $FAO_TT_OUT('old patch area address:  !XL', .PAT$GL_PATAREA[DSC$A_POINTER]);
2
2       !++
2       ! Check for conflicting patch area requests and set up alignment factor.
2       ! The alignment factor is set to the number of bytes in a longword, word,
2       ! byte, page, or quadword.
2       IF .PAT$GL_CONTEXT[ALIGN_BYTE]
2       THEN
2               ALIGN_FACTOR = A_BYTE;
2       IF .PAT$GL_CONTEXT[ALIGN_WORD]
2       THEN
2               ALIGN_FACTOR = A_WORD;
2       IF .PAT$GL_CONTEXT[ALIGN_LONG]
2       THEN
2               ALIGN_FACTOR = A_LONGWORD;
2       IF .PAT$GL_CONTEXT[ALIGN_QUAD]
2       THEN
2               ALIGN_FACTOR = A_QUADWORD;
2       IF .PAT$GL_CONTEXT[ALIGN_PAGE]
2       THEN
2               ALIGN_FACTOR = A_PAGE;
2
2       !++
2       ! Now round up image header patch area address and alter patch area
2       ! size to reflect any lost bytes.
2       !--
2       PATCH_AREA_ADR = ((.PAT$GL_PATAREA[DSC$A_POINTER] + (.ALIGN_FACTOR-1))/.ALIGN_FACTOR) * .ALIGN_FACTOR;
3       IF (.PATCH_AREA_ADR NEQA .PAT$GL_PATAREA[DSC$A_POINTER])                ! If rounding actually occurred
3               OR (.PAT$GL_PATAREA[DSC$W_LENGTH] EQL 0)                        ! or no patch space exists
2       THEN
3               BEGIN
3               PATCH_AREA_SIZ = .PAT$GL_PATAREA[DSC$W_LENGTH] +
3                               .PAT$GL_PATAREA[DSC$A_POINTER] - .PATCH_AREA_ADR;
4               IF (.PATCH_AREA_SIZ LEQ 0)                                      ! Check no patch area left
3               THEN
4                       BEGIN
5                       IF (.PAT$GL_PATAREA[DSC$A_POINTER] EQLA .PAT$GL_IHPPTR[IHP$L_RW_PATADR])
4                       THEN
4                               PAT$EXP_AREA (ONE_BLOCK)                        ! Get another block
4                       ELSE
4                               SIGNAL(PAT$_NOPATAREA, 2, .PAT$GL_PATAREA[DSC$A_POINTER],
4                                       .PAT$GL_PATAREA[DSC$W_LENGTH]);
6                       PATCH_AREA_ADR = ((.PAT$GL_PATAREA[DSC$A_POINTER] +
4                               (.ALIGN_FACTOR-1))/.ALIGN_FACTOR) * .ALIGN_FACTOR;
4                       PATCH_AREA_SIZ = .PAT$GL_PATAREA[DSC$W_LENGTH] +
4                                       .PAT$GL_PATAREA[DSC$A_POINTER] - .PATCH_AREA_ADR;
```

```
3                  END;
3          PAT$GL_PATAREA[DSC$A_POINTER] = .PATCH_AREA_ADR;            ! Set rounded address in header
3          PAT$GL_PATAREA[DSC$W_LENGTH] = .PATCH_AREA_SIZ;             ! Set rounded size in header
2          END;
2
2  !++
2  ! Output current patch area after alignment.
2  !--
2  $FAO_TT_OUT('new patch area size:      !XL', .PAT$GL_PATAREA[DSC$W_LENGTH]);
2  $FAO_TT_OUT('new patch area address:   !XL', .PAT$GL_PATAREA[DSC$A_POINTER]);
2
2  !++
2  ! Now enter the symbol into the user-defined symbol table with a value equal
2  ! to the aligned patch area address.
2  !--
2  !SYM_ENTRY_PTR = PAT$FIND_SYM(.LIST_ELEM_EXP1(.PAT$GL_HEAD_LST)); ! Check for previously defined symbol
2  !IF .SYM_ENTRY_PTR NEQA 0                                          ! Yes, was previously defined
2  !THEN                                                               ! Output informational message
2  !        BEGIN
2  !        SIGNAL(PAT$_REDEFSYM, 4, .SYM_CHCOUNT(.SYM_ENTRY_PTR), SYM_NAME(.SYM_ENTRY_PTR),
2  !                .SYM_VALUE(.SYM_ENTRY_PTR), .PATCH_AREA_ADR);
2  !        SYM_VALUE(.SYM_ENTRY_PTR) = .PATCH_AREA_ADR;               ! Set new value
2  !        END
2  !ELSE
2  TEMP_SYMTB = .PAT$GL_SYMTBPTR;
2  PAT$GL_SYMTBPTR = .PAT$GL_SYMHEAD;
2  PAT$DEFINE_SYM(.LIST_ELEM_EXP1(.PAT$GL_HEAD_LST), .PATCH_AREA_ADR, TRUE); ! Enter into list
2  PAT$GL_SYMTBPTR = .TEMP_SYMTB;
2
2
2  RETURN;
1  END;                                                                 ! End of PAT$ALIGN_CMD


                                                          .TITLE  PATSPA
                                                          .IDENT  \V04-000\

                                                          .PSECT  _PAT$PLIT,NOWRT,NOEXE,0

                                               1C  00000 P.AAA:   .BYTE   28
20  61  65  72  61  20  68  63  74  61  70  20  64  6C  6F  00001           .ASCII  \old patch area size:      !XL\        ;
        4C  58  21  20  20  20  20  20  3A  65  7A  69  73  00010
                                               1C  0001D P.AAB:   .BYTE   28
20  61  65  72  61  20  68  63  74  61  70  20  64  6C  6F  0001E           .ASCII  \old patch area address:   !XL\        ;
        4C  58  21  20  20  3A  73  73  65  72  64  64  61  0002D
                                               1C  0003A P.AAC:   .BYTE   28
20  61  65  72  61  20  68  63  74  61  70  20  77  65  6E  0003B           .ASCII  \new patch area size:      !XL\        ;
        4C  58  21  20  20  20  20  20  3A  65  7A  69  73  0004A
                                               1C  00057 P.AAD:   .BYTE   28
20  61  65  72  61  20  68  63  74  61  70  20  77  65  6E  00058           .ASCII  \new patch area address:   !XL\        ;
        4C  58  21  20  20  3A  73  73  65  72  64  64  61  00067

                                                          .PSECT  _PAT$OWN,NOEXE,2

                             41  41  50  03  00000 PAT_AREA_NAME:
                                                          .ASCII  <3>\PAA\                               ;
                                   00000003  00004 PA_NAME_DSC:
```

```
PATSPA                                       H 14
V04-000                                      16-Sep-1984 00:57:14    VAX-11 Bliss-32 V4.0-742              Page  10
                                             14-Sep-1984 12:52:47    DISK$VMSMASTER:[PATCH.SRC]PATSPA.B32;1     (3)

                                                        .LONG    3                                          ;
                          00000000' 00008               .ADDRESS PAT_AREA_NAME+1                            ;

                                             ISE$C_SIZE==        20
                                             TXT$C_SIZE==        4
                                             PAL$C_SIZE==        16
                                             ASD$C_SIZE==        9
                                             FWR$C_SIZE==        24
                                                        .EXTRN   PAT$FAO_OUT, PAT$GL_PAL_LHD
                                                        .EXTRN   PAT$GL_ERRCODE, PAT$GL_CONTEXT
                                                        .EXTRN   PAT$GL_FLAGS, PAT$GL_IMGHDR
                                                        .EXTRN   PAT$GL_PATAREA, PAT$GL_IHPPTR
                                                        .EXTRN   PAT$GL_ISELHD, PAT$GL_ISETAIL
                                                        .EXTRN   PAT$GL_NEWVPNMX
                                                        .EXTRN   PAT$GL_NEWVBNMX
                                                        .EXTRN   PAT$GL_IMGBLKS, PAT$GL_ISVADDR
                                                        .EXTRN   PAT$GL_HEAD_LST
                                                        .EXTRN   PAT$GL_SYMTBPTR
                                                        .EXTRN   PAT$GL_SYMHEAD, PAT$ALLOBLK
                                                        .EXTRN   PAT$CREMAP, PAT$DEFINE_SYM
                                                        .EXTRN   PAT$FIND_SYM, PAT$FREEZ
                                                        .EXTRN   PAT$MAP_ADDR, PAT$_CLOSEIN
                                                        .EXTRN   PAT$_CLOSEOUT, PAT$_OPENIN
                                                        .EXTRN   PAT$_OPENOUT, PAT$_READERR
                                                        .EXTRN   PAT$_SYSERROR, PAT$_WRITEERR
                                                        .WEAK    ACCESS_CHECK

                                                        .PSECT   _PAT$CODE,NOWRT,2

                               07FC 00000               .ENTRY   PAT$ALIGN_CMD, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1356
                                                                 R9,R10                                     ;
                5A 00000000G   EF  9E 00002             MOVAB    PAT$GL_SYMTBPTR, R10                       ;
                59 00000000G   EF  9E 00009             MOVAB    PAT$FAO_OUT, R9                            ;
                58 00000000'   EF  9E 00010             MOVAB    P.AAA, R8                                  ;
                57 00000000G   EF  9E 00017             MOVAB    PAT$GL_CONTEXT, R7                         ;
                56 00000000G   EF  9E 0001E             MOVAB    PAT$GL_PATAREA, R6                         ;
                7E         00  B6  3C 00025             MOVZWL   @PAT$GL_PATAREA, -(SP)                     ; 1423
                               58  DD 00029             PUSHL    R8                                         ;
                69             02  FB 0002B             CALLS    #2, PAT$FAO_OUT                            ;
                50             66  D0 0002E             MOVL     PAT$GL_PATAREA, R0                         ; 1424
                           04  A0  DD 00031             PUSHL    4(R0)                                      ;
                           1D  A8  9F 00034             PUSHAB   P.AAB                                      ;
                69             02  FB 00037             CALLS    #2, PAT$FAO_OUT                            ;
        03      67             06  E1 0003A             BBC      #6, PAT$GL_CONTEXT, 1$                     ; 1430
                53             01  D0 0003E             MOVL     #1, ALIGN_FACTOR                           ; 1432
        03      67             04  E1 00041 1$:         BBC      #4, PAT$GL_CONTEXT, 2$                     ; 1433
                53             02  D0 00045             MOVL     #2, ALIGN_FACTOR                           ; 1435
        03      67             02  E1 00048 2$:         BBC      #2, PAT$GL_CONTEXT, 3$                     ; 1436
                53             04  D0 0004C             MOVL     #4, ALIGN_FACTOR                           ; 1438
        03      67             03  E1 0004F 3$:         BBC      #3, PAT$GL_CONTEXT, 4$                     ; 1439
                53             08  D0 00053             MOVL     #8, ALIGN_FACTOR                           ; 1441
        05      67             05  E1 00056 4$:         BBC      #5, PAT$GL_CONTEXT, 5$                     ; 1442
                53       0200  8F  3C 0005A             MOVZWL   #512, ALIGN_FACTOR                         ; 1444
                52             66  D0 0005F 5$:         MOVL     PAT$GL_PATAREA, R2                         ; 1450
                50         04  A2  D0 00062             MOVL     4(R2), R0                                  ;
                51      FF A340    9E 00066             MOVAB    -1(ALIGN_FACTOR)[R0], R1                   ;
                51             53  C6 0006B             DIVL2    ALIGN_FACTOR, R1                           ;
        55          51              53  C5 0006E          MULL3   ALIGN_FACTOR, R1, PATCH_AREA_ADR                  :
                    50              55  D1 00072          CMPL    PATCH_AREA_ADR, R0                                : 1451
                                    04  12 00075          BNEQ    6$                                                :
                                    62  B5 00077          TSTW    (R2)                                              : 1452
                                    5D  12 00079          BNEQ    10$                                               :
                    52              62  3C 0007B 6$:      MOVZWL  (R2), R2                                          : 1455
        51          52              50  C1 0007E          ADDL3   R0, R2, R1                                        :
        54          51              55  C3 00082          SUBL3   PATCH_AREA_ADR, R1, PATCH_AREA_SIZ                : 1456
                                    46  14 00086          BGTR    9$                                                : 1457
                    51 00000000G    EF  D0 00088          MOVL    PAT$GL_IHPPTR, R1                                 : 1460
              14    A1              50  D1 0008F          CMPL    R0, 20(R1)                                        :
                                    0B  12 00093          BNEQ    7$                                                :
                                    01  DD 00095          PUSHL   #1                                                : 1462
           00000000V  EF            01  FB 00097          CALLS   #1, PAT$EXP_AREA                                  :
                                    11  11 0009E          BRB     8$                                                :
                                    05  BB 000A0 7$:      PUSHR   #^M<R0,R2>                                        : 1464
                                    02  DD 000A2          PUSHL   #2                                                :
                      006D811A      8F  DD 000A4          PUSHL   #7176474                                          :
           00000000G  00            04  FB 000AA          CALLS   #4, LIB$SIGNAL                                    :
                    50              66  D0 000B1 8$:      MOVL    PAT$GL_PATAREA, R0                                : 1466
        51          53        04    A0  C1 000B4          ADDL3   4(R0), ALIGN_FACTOR, R1                           : 1467
                                    51  D7 000B9          DECL    R1                                                : 1466
                    51              53  C6 000BB          DIVL2   ALIGN_FACTOR, R1                                  : 1467
        55          51              53  C5 000BE          MULL3   ALIGN_FACTOR, R1, PATCH_AREA_ADR                  :
                    51              60  3C 000C2          MOVZWL  (R0), R1                                          : 1469
        50          51        04    A0  C1 000C5          ADDL3   4(R0), R1, R0                                     :
        54          50              55  C3 000CA          SUBL3   PATCH_AREA_ADR, R0, PATCH_AREA_SIZ                :
                    50              66  D0 000CE 9$:      MOVL    PAT$GL_PATAREA, R0                                : 1471
              04    A0              55  D0 000D1          MOVL    PATCH_AREA_ADR, 4(R0)                             :
                    60              54  B0 000D5          MOVW    PATCH_AREA_SIZ, (R0)                              : 1472
                    7E        00    B6  3C 000D8 10$:     MOVZWL  @PAT$GL_PATAREA, -(SP)                            : 1478
                              3A    A8  9F 000DC          PUSHAB  P.AAC                                             :
                    69              02  FB 000DF          CALLS   #2, PAT$FAO_OUT                                   :
                    50              66  D0 000E2          MOVL    PAT$GL_PATAREA, R0                                : 1479
                              04    A0  DD 000E5          PUSHL   4(R0)                                             :
                              57    A8  9F 000E8          PUSHAB  P.AAD                                             :
                    69              02  FB 000EB          CALLS   #2, PAT$FAO_OUT                                   :
                    53              6A  D0 000EE          MOVL    PAT$GL_SYMTBPTR, TEMP_SYMTB                       : 1494
                    6A 00000000G    EF  D0 000F1          MOVL    PAT$GL_SYMHEAD, PAT$GL_SYMTBPTR                   : 1495
                                    01  DD 000F8          PUSHL   #1                                                : 1496
                                    55  DD 000FA          PUSHL   PATCH_AREA_ADR                                    :
                    50 00000000G    EF  D0 000FC          MOVL    PAT$GL_HEAD_LST, R0                               :
                              04    A0  DD 00103          PUSHL   4(R0)                                             :
           00000000G  EF            03  FB 00106          CALLS   #3, PAT$DEFINE_SYM                                :
                    6A              53  D0 0010D          MOVL    TEMP_SYMTB, PAT$GL_SYMTBPTR                       : 1497
                                        04 00110          RET                                                       : 1501

; Routine Size:  273 bytes,    Routine Base:  _PAT$CODE + 0000
```

```
GLOBAL ROUTINE PAT$BUILD_ISE (ISE_PTR,VPN,VBN,PAGE_CNT,ADR) : NOVALUE = ! Builds an ISD and enters it into I

!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine builds a new image section descriptor.  It is a normal
!       type image section with read-write, copy-on-reference attributes.
!       The virtual page number, virtual block number, and the page count
!       are input parameters.  The address of the image section table entry,
!       built around the image section descriptor, is returned.  The image
!       section entry is linked into the table.
!
! FORMAL PARAMETERS:
!
!       ISE_PTR - Pointer to image section entry built
!       VPN - Virtual page number of image section
!       VBN - Virtual block number of image section
!       PAGE_CNT - Number of pages in image section
!       ADR = [OPTIONAL] Address which is to be modified by the patch.
!
! IMPLICIT INPUTS:
!
!       The image section table is set up.
!
! IMPLICIT OUTPUTS:
!
!       A new image section descriptor is built.
!
! ROUTINE VALUE:
!
!       none
!
! COMPLETION CODES:
!
!       none
!
! SIDE EFFECTS:
!
!       If the ADR parameter is included in the call, we will propagate the
!       the image section attributes (of the image section containing the
!       address specified by ADR) to the newly created default patch area.
!
!--

BEGIN

BUILTIN
        NULLPARAMETER;

LOCAL
        PFC : BYTE,                                     ! Page Fault Cluster size
        TYPE : BYTE,                                    ! Type of image section
        FLAGS,                                          ! Image section Flags
        IDENT,                                          ! Image section Ident
        PREV_ISE_PTR : REF BLOCK[,BYTE],                ! Pointer to previous Image Section table en
        TEMP : REF BLOCK[,BYTE],                        ! Holds the FLINK from previous ISE
        LOCAL_ISE_PTR : REF BLOCK[,BYTE],               ! Image section table entry pointer
```

```
2               ISD_PTR : REF BLOCK[,BYTE];                                     ! Image section descriptor pointer
2
2       !++
2       ! Allocate space for new image section table entry.
2       ! ***** UNTIL SYSTEM IS UPDATED TO CONTAIN AN IDENT PERFORM TEST ON WHAT
2       ! ***** SIZE TO USE.
2       !--
2       IF PAT$K_LENPRIV GTR ISD$K_LENPRIV
2       THEN
2               PAT$ALLOBLK(ISE$C_SIZE+PAT$K_LENPRIV, .ISE_PTR)
2       ELSE
2               PAT$ALLOBLK(ISE$C_SIZE+ISD$K_LENPRIV, .ISE_PTR);
2
2       !++
2       ! Now link the new entry into the table.
2       ! This is accomplished by traversing the Image Section Table Entries, looking for any
2       ! Non-Based Global or Stack ISDs which follow the last "Normal" ISD.  When this location
2       ! is found, the links in the affected ISEs are modified to include the new PATCH ISE.
2       !--
2       LOCAL_ISE_PTR = .PAT$GL_ISELHD;                                         ! Get the list head.
2       PREV_ISE_PTR = .LOCAL_ISE_PTR;                                          ! Set PREV = Current for first pass.
2       ISD_PTR = CH$PTR (.LOCAL_ISE_PTR, ISE$C_SIZE);                          ! Point to the first ISD in the list.
2
3       UNTIL ( (.LOCAL_ISE_PTR EQL 0) OR
3               (.ISD_PTR[ISD$B_TYPE] EQL ISD$K_USRSTACK) OR
2               (.ISD_PTR[ISD$V_GBL] AND NOT .ISD_PTR[ISD$V_BASED]) ) DO
3               BEGIN
3               IF NOT NULLPARAMETER (5)                                        ! Was an address included in the call?
3               THEN                                                            ! If so, then check to see if it maps
3                   IF  .ADR GEQ .ISD_PTR[ISD$V_VPN] ^9                         ! into this ISD.
5                   AND .ADR LEQ ((.ISD_PTR[ISD$V_VPN] +
3                                 .ISD_PTR[ISD$W_PAGCNT]) ^9) - 1
3                   THEN                                                        ! It does map into this ISD, so save the
4                       BEGIN                                                   ! attributes for the new default patch area
4                       PFC =   .ISD_PTR [ISD$B_PFC];                           ! Page Fault Cluster size.
4                       FLAGS = .ISD_PTR [ISD$L_FLAGS];                         ! Image section Flags.
4                       TYPE =  .ISD_PTR [ISD$B_TYPE];                          ! Image section Type.
4                       IDENT = .ISD_PTR [ISD$L_IDENT];                         ! Image section Ident.
3                       END;
3               PREV_ISE_PTR = .LOCAL_ISE_PTR;                                  ! Save the address of the just checked ISE.
3               LOCAL_ISE_PTR = .LOCAL_ISE_PTR[ISE$L_NXTISE];                   ! Advance the pointer to the next ISE.
3               ISD_PTR = CH$PTR (.LOCAL_ISE_PTR, ISE$C_SIZE);                  ! Point to the next ISD also.
2               END;
2
2       !++
2       ! At this point we should be positioned to the location for inserting the new PATCH ISE/ISD pair.
2       !--
2       LOCAL_ISE_PTR = CH$PTR (..ISE_PTR, 0);                                  ! Pick up the address of the new ISE.
2       TEMP = .PREV_ISE_PTR[ISE$L_NXTISE];                                     ! Save the FLINK to next ISE.
2       PREV_ISE_PTR[ISE$L_NXTISE] = .LOCAL_ISE_PTR;                            ! Set FLINK to the new ISE.
2       LOCAL_ISE_PTR[ISE$L_NXTISE] = .TEMP;                                    ! Remember to point to the next ISE.
2
2       !++
2       ! Initialize the image section table information.
2       !--
2       LOCAL_ISE_PTR[ISE$L_MAPVST] = 0;
2       LOCAL_ISE_PTR[ISE$L_MAPVEND] = 0;
```

```
                                                       L 14
PATSPA                                                 16-Sep-1984 00:57:14    VAX-11 Bliss-32 V4.0-742                    Page  14
V04-000                                                14-Sep-1984 12:52:47    DISK$VMSMASTER:[PATCH.SRC]PATSPA.B32;1        (4)

;  466        1616  2 LOCAL_ISE_PTR[ISE$L_IMGVST] = 0;
;  467        1617  2 LOCAL_ISE_PTR[ISE$L_IMGVEND] = 0;
;  468        1618  2
;  469        1619  2 !++
;  470        1620  2 ! Now build the image section descriptor.
;  471        1621  2 !--
;  472        1622  2 ISD_PTR = CH$PTR(.LOCAL_ISE_PTR, ISE$C_SIZE);                            ! Point to ISD
;  473        1623  2 ! ****** THIS SHOULD CHANGE WHEN IDENT FIELD IS DEFINED FOR PROCESS PRIVATE IMAGE SECTIONS.
;  474        1624  2 ! ISD_PTR[ISD$W_SIZE] = (IF (PAT$K_LENPRIV GTR ISD$K_LENPRIV) THEN PAT$K_LENPRIV ELSE ISD$K_LENPRIV);
;  475        1625  2 ! *****
;  476        1626  2 ISD_PTR[ISD$W_SIZE] = ISD$K_LENPRIV;
;  477        1627  2 ISD_PTR[ISD$W_PAGCNT] = .PAGE_CNT;
;  478        1628  2 ISD_PTR[ISD$L_VPNPFC] = .VPN;
;  479        1629  2 ISD_PTR[ISD$B_PFC] = 0;
;  480        1630  2 ISD_PTR[ISD$L_FLAGS] = 0;
;  481        1631  2 ISD_PTR[ISD$V_CRF] = TRUE;
;  482        1632  2 ISD_PTR[ISD$V_WRT] = TRUE;
;  483        1633  2 ISD_PTR[ISD$V_MATCHCTL] = ISD$K_MATNEV;
;  484        1634  2 ISD_PTR[ISD$B_TYPE] = ISD$K_NORMAL;
;  485        1635  2 ISD_PTR[ISD$L_VBN] = .VBN;
;  486        1636  2 ISD_PTR[ISD$L_IDENT] = 0;
;  487        1637  2
;  488        1638  2 IF NOT NULLPARAMETER (5)                                                   ! Should we propagate the "patched"
;  489        1639  2 THEN                                                                       ! image section attributes?
;  490        1640  3     BEGIN
;  491        1641  3     ISD_PTR[ISD$B_PFC] = .PFC;
;  492        1642  3     ISD_PTR[ISD$L_FLAGS] = .FLAGS;
;  493        1643  3     ISD_PTR[ISD$B_TYPE] = .TYPE;
;  494        1644  3     ISD_PTR[ISD$L_IDENT] = .IDENT;
;  495        1645  2     END;
;  496        1646  2
;  497        1647  2 RETURN;
;  498        1648  1 END;                                                                       ! End of PAT$BUILD_ISE



                                        01FC 00000          .ENTRY  PAT$BUILD_ISE, Save R2,R3,R4,R5,R6,R7,R8      ; 1502
                              04    AC    DD 00002          PUSHL   ISE_PTR                                       ; 1568
                                    28    DD 00005          PUSHL   #40                                           ;
              00000000G    EF       02    FB 00007          CALLS   #2, PAT$ALLOBLK                               ;
                           51 00000000G   EF    D0 0000E    MOVL    PAT$GL_ISELHD, LOCAL_ISE_PTR                  ; 1578
                           53       51    D0 00015          MOVL    LOCAL_ISE_PTR, PREV_ISE_PTR                   ; 1579
                           50 14    A1    9E 00018 1$:      MOVAB   20(R1), ISD_PTR                               ; 1580
                                    51    D5 0001C          TSTL    LOCAL_ISE_PTR                                 ; 1582
                                    5B    13 0001E          BEQL    4$                                            ;
                     FD    8F 0B    A0    91 00020          CMPB    11(ISD_PTR), #253                             ; 1583
                                    54    13 00025          BEQL    4$                                            ;
                           05 08    A0    E9 00027          BLBC    8(ISD_PTR), 2$                                ; 1584
              4B     09    A0       01    E1 0002B          BBC     #1, 9(ISD_PTR), 4$                            ;
                           05       6C    91 00030 2$:      CMPB    (AP), #5                                      ; 1586
                                    3E    1F 00033          BLSSU   3$                                            ;
                              14    AC    D5 00035          TSTL    20(AP)                                        ;
                                    39    13 00038          BEQL    3$                                            ;
     52      04    A0              15       00    EF 0003A  EXTZV   #0, #21, 4(ISD_PTR), R2                       ; 1588
                   52              52       09    78 00040  ASHL    #9, R2, R2                                    ;
```

```
                              52     14  AC  D1 00044        CMPL    ADR, R2                                  ;
                                         29  19 00048        BLSS    3$                                       ;
         52      04   A0      15         00  EF 0004A        EXTZV   #0, #21, 4(ISD_PTR), R2                  ; 1590
                              58     02  A0  3C 00050        MOVZWL  2(ISD_PTR), R8                           ;
                              52         58  C0 00054        ADDL2   R8, R2                                   ;
                      52      52         09  78 00057        ASHL    #9, R2, R2                               ;
                                         52  D7 0005B        DECL    R2                                       ;
                              52     14  AC  D1 0005D        CMPL    ADR, R2                                  ;
                                         10  14 00061        BGTR    3$                                       ;
                              57     07  A0  90 00063        MOVB    7(ISD_PTR), PFC                          ; 1593
                              55     08  A0  D0 00067        MOVL    8(ISD_PTR), FLAGS                        ; 1594
                              56     0B  A0  90 0006B        MOVB    11(ISD_PTR), TYPE                        ; 1595
                              54     10  A0  D0 0006F        MOVL    16(ISD_PTR), IDENT                       ; 1596
                              53         51  D0 00073 3$:    MOVL    LOCAL_ISE_PTR, PREV_ISE_PTR              ; 1598
                              51         61  D0 00076        MOVL    (LOCAL_ISE_PTR), LOCAL_ISE_PTR           ; 1599
                                         9D  11 00079        BRB     1$                                       ; 1600
                              51     04  BC  D0 0007B 4$:    MOVL    @ISE_PTR, LOCAL_ISE_PTR                  ; 1606
                              52         63  D0 0007F        MOVL    (PREV_ISE_PTR), TEMP                     ; 1607
                              63         51  D0 00082        MOVL    LOCAL_ISE_PTR, (PREV_ISE_PTR)            ; 1608
                              61         52  D0 00085        MOVL    TEMP, (LOCAL_ISE_PTR)                    ; 1609
                                     0C  A1  7C 00088        CLRQ    12(LOCAL_ISE_PTR)                        ; 1614
                                     04  A1  7C 0008B        CLRQ    4(LOCAL_ISE_PTR)                         ; 1616
                              50     14  A1  9E 0008E        MOVAB   20(R1), ISD_PTR                          ; 1622
                              60         10  B0 00092        MOVW    #16, (ISD_PTR)                           ; 1626
                      02      A0     10  AC  B0 00095        MOVW    PAGE_CNT, 2(ISD_PTR)                     ; 1627
                      04      A0     08  AC  D0 0009A        MOVL    VPN, 4(ISD_PTR)                          ; 1628
                                     07  A0  94 0009F        CLRB    7(ISD_PTR)                               ; 1629
                              52     08  A0  9E 000A2        MOVAB   8(ISD_PTR), R2                           ; 1630
                                         62  D4 000A6        CLRL    (R2)                                     ;
                              62         0A  88 000A8        BISB2   #10, (R2)                                ; 1632
         62           03      04         03  F0 000AB        INSV    #3, #4, #3, (R2)                         ; 1633
                                     0B  A0  94 000B0        CLRB    11(ISD_PTR)                              ; 1634
                      0C      A0     0C  AC  D0 000B3        MOVL    VBN, 12(ISD_PTR)                         ; 1635
                                     10  A0  D4 000B8        CLRL    16(ISD_PTR)                              ; 1636
                              05         6C  91 000BB        CMPB    (AP), #5                                 ; 1638
                                         14  1F 000BE        BLSSU   5$                                       ;
                                     14  AC  D5 000C0        TSTL    20(AP)                                   ;
                                         0F  13 000C3        BEQL    5$                                       ;
                      07      A0         57  90 000C5        MOVB    PFC, 7(ISD_PTR)                          ; 1641
                              62         55  D0 000C9        MOVL    FLAGS, (R2)                              ; 1642
                      0B      A0         56  90 000CC        MOVB    TYPE, 11(ISD_PTR)                        ; 1643
                      10      A0         54  D0 000D0        MOVL    IDENT, 16(ISD_PTR)                       ; 1644
                                             04 000D4 5$:    RET                                              ; 1648

; Routine Size:  213 bytes,    Routine Base:  _PAT$CODE + 0111
```

```
1 GLOBAL ROUTINE PAT$EXP_AREA (NUM_BLKS, ADR) : NOVALUE =          ! Expands patch area
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine expands the read-write patch area defined in the image
1 !       header.  If there is no patch area, then an image section descriptor
1 !       is created for it.  If the image section which is being created is a
1 !       due to either an INSERT or REPLACE command then the attributes of the
1 !       image section are propagated to the new image section.  In either case,
1 !       the image header is updated to describe the expanded patch area.
1 !
1 !       If the patch area is mapped to the highest address used during this
1 !       patch session, then the patch area can be expanded contiguously.
1 !       In this case, the image section descriptor is updated to hold a new
1 !       page count and the patch area size in the image header is increased.
1 !       If the patch area is not the highest address used, then the patch area
1 !       must be relocated to another area, which will be contiguous.  This
1 !       involves expanding the program region, copying in the old patch area,
1 !       and then changing the image section table entry to point to a new
1 !       mapped address.  The image header and image section descriptor counts
1 !       are incremented as above.
1 !
1 !       NOTE:  The patch area must be mapped contiguously in order for
1 !       the mapping of addresses to work.  It could also be accomplished
1 !       if two image section table entries were created.  However, this
1 !       would require an extra, unneccessary image section descriptor.
1 !
1 !       Some of the PATCH commands which deposit symbolic instructions do an
1 !       PAT$EXPAREA just to force the address to be non-zero so that
1 !       the symbolic instruction encoder can correctly encode operands.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NUM_BLKS - Number of blocks to be allocated for the patch area
1 !       ADR -[OPT] The address which we will use to propagate the image section
1 !                   attributes.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       The image header and image section entry table must be set up.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       none
1 !
1 ! ROUTINE VALUE:
1 !
1 !       none
1 !
1 ! COMPLETION CODES:
1 !
1 !       none
1 !
1 ! SIDE EFFECTS:
1 !
1 !       A new patch area is set up.  The image header is updated to
```

```
 1 !        describe the new patch area.
 1 !
 1 !        ** If the file is being patched in absolute mode, we cannot
 1 !        expand the file (it would more than likely corrupt it!).
 1 !        In this instance, we'll abort the command back to the patch
 1 !        command prompt via an error severity signal.
 1 !
 1 !--
 1
 2 BEGIN
 2
 2 BUILTIN
 2          NULLPARAMETER;
 2
 2 LITERAL
 2          START_OFF = 0,                                          ! Offset to starting address
 2          END_OFF = 1;                                            ! Offset to ending address
 2
 2 LOCAL
 2          ISE_PTR : REF BLOCK[,BYTE],                             ! Pointer to image section table entry
 2          ISD_PTR : REF BLOCK[,BYTE],                             ! Pointer to image section descriptor
 2          MAPPED_ADDR;                                            ! Mapped address
 2
 2 IF .PAT$GL_FLAGS [PAT$S_ABSOLUTE]                                 ! If we're in absolute mode, then someone ha
 2 THEN     SIGNAL (PAT$_DATTOOLNG);                                 ! us too long a datum, and the file was expe
 2                                                                  ! to be expanded (a no no!), return to promp
 2
 2 !++
 2 ! If this is a non-PIC shareable image we do not expand the patch area to protect images
 2 ! previously linked against having inconsistent Global Section Descriptors.  Else, if it
 2 ! is a PIC shareable image, we may without reservation, expand the patch area.
 2 !--
 3 IF ((.PAT$GL_IMGHDR[IHD$B_IMGTYPE] EQLU IHD$K_LIM) AND (NOT .PAT$GL_IMGHDR[IHD$V_PICIMG]))
 2 THEN
 2          SIGNAL(PAT$_EXPSHRPAT+MSG$K_SEVERE);
 2
 2 !++
 2 ! If there is no patch area defined yet, then build an image section table
 2 ! entry and an image section descriptor for it.
 2 !--
 3 IF (.PAT$GL_IHPPTR[IHP$L_RW_PATADR] EQLA 0)
 2 THEN
 3          BEGIN
 3          !++
 3          ! Build an Image Section table entry as no Patch Area was defined.
 3          !--
 3          IF NULLPARAMETER (2)
 3          THEN PAT$BUILD_ISE(ISE_PTR, .PAT$GL_NEWVPNMX+1, .PAT$GL_NEWVBNMX+1, .NUM_BLKS)
 3          ELSE PAT$BUILD_ISE(ISE_PTR, .PAT$GL_NEWVPNMX+1, .PAT$GL_NEWVBNMX+1, .NUM_BLKS, .ADR);
 3          ISD_PTR = CH$PTR(.ISE_PTR, ISE$C_SIZE);
 3          END
 2 ELSE
 3          BEGIN
 3          !++
 3          ! Find the image section table entry which describes the patch area.
 3          !--
 3          PAT$MAP_ADDR(.PAT$GL_IHPPTR[IHP$L_RW_PATADR], MAPPED_ADDR, ISE_PTR);
```

```
        ISD_PTR = CH$PTR(.ISE_PTR, ISE$C_SIZE);
        ISD_PTR[ISD$W_PAGCNT] = .ISD_PTR[ISD$W_PAGCNT] + .NUM_BLKS; ! Expand size of image section
        END;

!++
! Update the VPN and VBN for the last ones used in the new image for
! the image section.
!--
PAT$GL_NEWVPNMX = .PAT$GL_NEWVPNMX + .NUM_BLKS;
PAT$GL_NEWVBNMX = .PAT$GL_NEWVBNMX + .NUM_BLKS;

!++
! Now create the patch area, i.e., map it into the image.  This is done
! with an expand region instead of a create and map as the area is not defined
! in the old image.
!--
PAT$GL_ERRCODE = $EXPREG(PAGCNT = .ISD_PTR[ISD$W_PAGCNT]
                        , RETADR = PAT$GL_ISVADDR);
IF NOT .PAT$GL_ERRCODE
THEN
        SIGNAL(PAT$_SYSERROR, 0, .PAT$GL_ERRCODE);

!++
! If the patch area was expanded, and not created, then copy in the old
! patch area part.
!--
IF (.ISD_PTR[ISD$W_PAGCNT] NEQ .NUM_BLKS)
THEN
        CH$MOVE((.ISD_PTR[ISD$W_PAGCNT] - .NUM_BLKS) * A_PAGE,
                .ISE_PTR[ISE$L_MAPVST], .PAT$GL_ISVADDR[START_OFF]);

!++
! Initialize the image section table entry.
!--
ISE_PTR[ISE$L_MAPVST] = .PAT$GL_ISVADDR[START_OFF];
ISE_PTR[ISE$L_MAPVEND] = .PAT$GL_ISVADDR[END_OFF];
ISE_PTR[ISE$L_IMGVST] = .ISD_PTR[ISD$V_VPN] ^9;
ISE_PTR[ISE$L_IMGVEND] = ((.ISD_PTR[ISD$V_VPN] + .ISD_PTR[ISD$W_PAGCNT]) ^9) - 1;

!++
! Increment the number of blocks in the new image.
!--
PAT$GL_IMGBLKS = .PAT$GL_IMGBLKS + .NUM_BLKS;

!++
! Update the patch area descriptor in the image header.
!--
PAT$GL_PATAREA[DSC$W_LENGTH] = .PAT$GL_PATAREA[DSC$W_LENGTH] + (.NUM_BLKS * A_PAGE);
IF (.PAT$GL_PATAREA[DSC$A_POINTER] EQLA 0)
THEN
        PAT$GL_PATAREA[DSC$A_POINTER] = .ISE_PTR[ISE$L_IMGVST];

!++
! Now update the patch area list entry for the default patch area.
!--
PAT$ADD_PAL(.ISE_PTR[ISE$L_IMGVST], .ISE_PTR[ISE$L_IMGVEND], PAL$K_EXP_PAREA);
```

```
;  671      1820  2 RETURN;
;  672      1821  2
;  673      1822  1 END;                                                         ! END OF PAT$EXP_AREA


                                                        .EXTRN  SYS$EXPREG

                                   0FFC 00000           .ENTRY  PAT$EXP_AREA, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1649
                                                                R10,R11                                       :
                 5B 00000000G  EF  9E 00002             MOVAB   PAT$GL_NEWVBNMX, R11                          :
                 5A 00000000G  EF  9E 00009             MOVAB   PAT$GL_ERRCODE, R10                           :
                 59 00000000G  EF  9E 00010             MOVAB   PAT$GL_ISVADDR, R9                            :
                 58 00000000G  00  9E 00017             MOVAB   LIB$SIGNAL, R8                                :
                 5E            08  C2 0001E             SUBL2   #8, SP                                        :
         09 00000000G  EF      06  E1 00021             BBC     #6, PAT$GL_FLAGS, 1$                          ; 1729
                006D80A2  8F   DD 00029                 PUSHL   #7176354                                      ; 1730
                 68            01  FB 0002F             CALLS   #1, LIB$SIGNAL                                :
                 50 00000000G  EF  D0 00032 1$:         MOVL    PAT$GL_IMGHDR, R0                             ; 1738
                 02        11  A0  91 00039             CMPB    17(R0), #2                                    :
                               0E  12 0003D             BNEQ    2$                                            :
         09          20  A0    03  E0 0003F             BBS     #3, 32(R0), 2$                                :
                006D82D2  8F   DD 00044                 PUSHL   #7176914                                      ; 1740
                 68            01  FB 0004A             CALLS   #1, LIB$SIGNAL                                :
                 57        04  AC  D0 0004D 2$:         MOVL    NUM_BLKS, R7                                  ; 1753
                 50 00000000G  EF  D0 00051             MOVL    PAT$GL_IHPPTR, R0                             ; 1746
                           14  A0  D5 00058             TSTL    20(R0)                                        :
                               39  12 0005B             BNEQ    6$                                            :
         51            6B      01  C1 0005D             ADDL3   #1, PAT$GL_NEWVBNMX, R1                       ; 1753
         50 00000000G  EF      01  C1 00061             ADDL3   #1, PAT$GL_NEWVPNMX, R0                       :
         02            6C      91 00069                 CMPB    (AP), #2                                      ; 1752
                               05  1F 0006C             BLSSU   3$                                            :
                           08  AC  D5 0006E             TSTL    8(AP)                                         :
                               0E  12 00071             BNEQ    4$                                            :
                        0083   8F  BB 00073 3$:         PUSHR   #^M<R0,R1,R7>                                 ; 1753
                          0C   AE  9F 00077             PUSHAB  ISE_PTR                                       :
               FEAC  CF        04  FB 0007A             CALLS   #4, PAT$BUILD_ISE                             :
                               0F  11 0007F             BRB     5$                                            :
                          08   AC  DD 00081 4$:         PUSHL   ADR                                           ; 1754
                        0083   8F  BB 00084             PUSHR   #^M<R0,R1,R7>                                 :
                          10   AE  9F 00088             PUSHAB  ISE_PTR                                       :
               FE9B  CF        05  FB 0008B             CALLS   #5, PAT$BUILD_ISE                             :
         56            6E      14  C1 00090 5$:         ADDL3   #20, ISE_PTR, ISD_PTR                         ; 1755
                               17  11 00094             BRB     7$                                            ; 1746
                               5E  DD 00096 6$:         PUSHL   SP                                            ; 1762
                          08   AE  9F 00098             PUSHAB  MAPPED_ADDR                                   :
                          14   A0  DD 0009B             PUSHL   20(R0)                                        :
            00000000G  EF      03  FB 0009E             CALLS   #3, PAT$MAP_ADDR                              :
         56            6E      14  C1 000A5             ADDL3   #20, ISE_PTR, ISD_PTR                         ; 1763
                  02   A6      57  A0 000A9             ADDW2   R7, 2(ISD_PTR)                                ; 1764
            00000000G  EF      57  C0 000AD 7$:         ADDL2   R7, PAT$GL_NEWVPNMX                           ; 1771
                       6B      57  C0 000B4             ADDL2   R7, PAT$GL_NEWVBNMX                           ; 1772
                       7E      7C 000B7                 CLRQ    -(SP)                                         ; 1780
                               59  DD 000B9             PUSHL   R9                                            :
                       7E  02  A6  3C 000BB             MOVZWL  2(ISD_PTR), -(SP)                             :
            00000000G  00      04  FB 000BF             CALLS   #4, SYS$EXPREG                                :
                       6A      50  D0 000C6             MOVL    R0, PAT$GL_ERRCODE                            :
```

```
                                 0D          6A  E8 000C9          BLBS     PAT$GL_ERRCODE, 8$                              ; 1781
                                             6A  DD 000CC          PUSHL    PAT$GL_ERRCODE                                  ; 1783
                                             7E  D4 000CE          CLRL     -(SP)                                           ;
                            00000000G        8F  DD 000D0          PUSHL    #PAT$_SYSERROR                                  ;
                                 68          03  FB 000D6          CALLS    #3, LIB$SIGNAL                                  ;
          57      02      A6     10          00  ED 000D9 8$:      CMPZV    #0, #16, 2(ISD_PTR), R7                         ; 1789
                                 14          13  000DF             BEQL     9$                                              ;
                                 51     02   A6  3C 000E1          MOVZWL   2(ISD_PTR), R1                                  ; 1791
                                 51          57  C2 000E5          SUBL2    R7, R1                                          ;
                          51     51          09  78 000E8          ASHL     #9, R1, R1                                      ;
                                 50          6E  D0 000EC          MOVL     ISE_PTR, R0                                     ; 1792
                  00      B9     0C    B0    51  28 000EF          MOVC3    R1, @12(R0), @PAT$GL_ISVADDR                    ;
                                 50          6E  D0 000F5 9$:      MOVL     ISE_PTR, R0                                     ; 1797
                                 0C    A0    69  7D 000F8          MOVQ     PAT$GL_ISVADDR, 12(R0)                          ;
          51      04      A6     15          00  EF 000FC          EXTZV    #0, #21, 4(ISD_PTR), R1                         ; 1799
                  04      A0     51          09  78 00102          ASHL     #9, R1, 4(R0)                                   ;
          51      04      A6     15          00  EF 00107          EXTZV    #0, #21, 4(ISD_PTR), R1                         ; 1800
                                 56     02   A6  3C 0010D          MOVZWL   2(ISD_PTR), R6                                  ;
                                 56          51  C0 00111          ADDL2    R1, R6                                          ;
                          56     56          09  78 00114          ASHL     #9, R6, R6                                      ;
                          08     A0     FF   A6  9E 00118          MOVAB    -1(R6), 8(R0)                                   ;
                    00000000G    EF          57  C0 0011D          ADDL2    R7, PAT$GL_IMGBLKS                              ; 1805
                                 51 00000000G EF D0 00124          MOVL     PAT$GL_PATAREA, R1                              ; 1810
                          52     57          09  78 0012B          ASHL     #9, R7, R2                                      ;
                                 61          52  A0 0012F          ADDW2    R2, (R1)                                        ;
                                        04   A1  D5 00132          TSTL     4(R1)                                           ; 1811
                                 05          12  00135             BNEQ     10$                                             ;
                          04     A1     04   A0  D0 00137          MOVL     4(R0), 4(R1)                                    ; 1813
                                 01          DD  0013C 10$:        PUSHL    #1                                              ; 1818
                                 7E     04   A0  7D 0013E          MOVQ     4(R0), -(SP)                                    ;
                    00000000V    EF          03  FB 00142          CALLS    #3, PAT$ADD_PAL                                 ;
                                             04  00149             RET                                                      ; 1822

; Routine Size:  330 bytes,    Routine Base:  _PAT$CODE + 01E6
```

```
GLOBAL ROUTINE PAT$ADD_PAL (START_ADR, END_ADR, PAT_AREA_FLAG) : NOVALUE =      ! EXPANDS PATCH AREAS

!++
! FUNCTIONAL DESCRIPTION:
!
!       THIS ROUTINE MAINTAINS THE PATCH AREA LIST (PAL).  THIS INCLUDES
!       UPDATING THE ENTRY FOR THE DEFAULT PATCH AREA WHENEVER PATCH EXPANDS
!       IT AND CREATING ENTRIES WHENEVER THE USER ISSUES A "SET PATCH_AREA"
!       COMMAND.  THE FIRST ENTRY ON THE LIST IS ALWAYS THE DEFAULT PATCH AREA.
!
!       THE PATCH AREA LIST IS USED TO CORRECTLY OUTPUT ADDRESSES FOR
!       PATCH AREA TO THE OUTPUT COMMAND FILE.  THESE ADDRESSES MUST BE
!       WRITTEN TO THE FILE AS SYMBOLIC NAMES PLUS OFFSETS BECAUSE THE
!       IMAGES IN THE FIELD MAY HAVE BEEN PATCHED BY CUSTOMERS ( THUS
!       CHANGING THE NEXT FREE PATCH AREA ADDRESS ).  BY OUTPUTING PATCH
!       AREA ADDRESSES AS SYMBOLIC NAMES, PATCH WILL PERMIT PATCHES TO
!       USE DIFFERENT PATCH AREA ADDRESSES.
!
!       AN ENTRY IN THE PATCH AREA LIST HAS THE FOLLOWING FORMAT:
!
!               +------------------------------+
!               |        FORWARD LINK          |  PAL$L_FLINK
!               +------------------------------+
!               |      STARTING ADDRESS        |  PAL$L_ST_ADR
!               +------------------------------+
!               |       ENDING ADDRESS         |  PAL$L_END_ADR
!               +------------------------------+
!               |      PATCH AREA NAME         |  PAL$L_CS_NAME
!               +------------------------------+
!
!       THE PATCH AREA NAME CONSISTS OF AN ASCIC STRING, WHICH IS ALWAYS A
!       COUNT OF THREE FOLLOWED BY THE ASCII CHARACTERS "P", "A", AND A THIRD
!       CHARACTER RANGING FROM "A" TO "Z".  THIS NAME IS USED TO OUTPUT
!       SYMBOLIC REFERENCES TO THE OUTPUT COMMAND FILE FOR ALL ADDRESSES WITHIN
!       THE PATCH AREAS INSTEAD OF ABSOLUTE VALUES.
!
!       THIS ROUTINE ALSO CAUSES A SYMBOL TO BE DEFINED FOR THE STARTING ADDRESS
!       OF THE PATCH AREA.
!
! FORMAL PARAMETERS:
!
!       START_ADR - STARTING ADDRESS OF THE PATCH AREA
!       END_ADR - ENDING ADDRESS OF THE PATCH AREA
!       PAT_AREA_FLAG - INDICATOR FOR TYPE OF PAL UPDATE
!                       PAL$K_EXP_PAREA = 1  - EXPANDING DEFAULT PATCH AREA
!                       PAL$K_ADD_PAREA = 0  - ADDING NEW PATCH AREA ENTRY
!
! IMPLICIT INPUTS:
!
!       THE FREE STORAGE ROUTINES MUST HAVE BEEN INITIALIZED.
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
!
```

```
 1 !        NONE
 1 !
 1 ! COMPLETION CODES:
 1 !
 1 !        NONE
 1 !
 1 ! SIDE EFFECTS:
 1 !
 1 !        THE PATCH AREA LIST IS UPDATED.  EITHER AN ENTRY IS MODIFIED OR
 1 !        A NEW LINK IS CREATED.  IN THE LATTER CASE, THE NEXT PATCH AREA NAME
 1 !        IS ALSO UPDATED.  THE NEXT PATCH AREA NAME IS ALSO UPDATED.
 1 !
 1 !--
 1
 2 BEGIN
 2
 2 LOCAL
 2        TEMP_SYMTB,                                            ! Temporary symbol table pointer
 2        NEW_PTR : REF BLOCK[,BYTE],                            ! POINTER TO NEW PAL ENTRY
 2        TEMP_PTR : REF BLOCK[,BYTE],                           ! POINTER TO CURRENT PAL ENTRY
 2        NAME_DESC : BLOCK[8,BYTE];                             ! STRING DESCRIPTOR FOR DEFAULT PATCH AREA N
 2
 2 !++
 2 ! FIRST, LOOP THROUGH THE PATCH AREA LIST TRYING TO FIND AN ENTRY FOR THIS
 2 ! PATCH AREA, I.E., HAS THIS PATCH AREA JUST BEEN EXPANDED.  IF SO, UPDATE
 2 ! THE PAL ENTRY AND RETURN.  IF NOT, FALL THROUGH TO CREATE A NEW PAL ENTRY.
 2 !--
 2 TEMP_SYMTB = .PAT$GL_SYMTBPTR;                                ! Remember current label symbol table
 2 IF (TEMP_PTR = CH$PTR(.PAT$GL_PAL_LHD, 0)) NEQ 0              ! GET FIRST ENTRY IN LIST
 2 THEN
 2        REPEAT
 3                BEGIN
 3                !++
 3                ! IF THE DEFAULT PATCH AREA WAS CREATED, THEN BOTH THE STARTING
 3                ! AND ENDING ADDRESSES MUST BE RESET.  IF THE DEFAULT PATCH
 3                ! AREA WAS EXPANDED, THEN THE STARTING ADDRESS REMAINS THE
 3                ! SAME AND THE ENDING ADDRESS IS UPDATED.  THIS WILL NEED
 3                ! SOME NEW INVENTION WHEN READ-ONLY PATCH AREAS ARE
 3                ! ALSO ADDED.
 3                !--
 3                IF .PAT_AREA_FLAG EQL PAL$K_EXP_PAREA
 3                THEN
 4                        BEGIN
 4                        TEMP_PTR[PAL$L_END_ADR] = .END_ADR;
 4                        IF .TEMP_PTR[PAL$L_START_ADR] EQLA 0
 4                        THEN
 5                                BEGIN
 5                                TEMP_PTR[PAL$L_START_ADR] = .START_ADR;
 5                                NAME_DESC[DSC$W_LENGTH] = .PAT_AREA_NAME[0];
 5                                NAME_DESC[DSC$A_POINTER] = CH$PTR(TEMP_PTR[PAL$L_CS_NAME], 1);
 5                                PAT$GL_SYMTBPTR = .PAT$GL_SYMHEAD;
 5                                PAT$DEFINE_SYM(NAME_DESC, .START_ADR, FALSE);
 5                                PAT$GL_SYMTBPTR = .TEMP_SYMTB;
 4                                END;
 4                        RETURN;
 3                        END;
 3                IF (.START_ADR GEQA .TEMP_PTR[PAL$L_START_ADR]) AND
```

```
;  789        1937 4                    (.END_ADR EQLA .TEMP_PTR[PAL$L_END_ADR])
;  790        1938 3                THEN
;  791        1939 3                        RETURN;
;  792        1940 3                IF .TEMP_PTR[PAL$L_FLINK] NEQA 0
;  793        1941 3                THEN
;  794        1942 3                        TEMP_PTR = .TEMP_PTR[PAL$L_FLINK]
;  795        1943 3                ELSE
;  796        1944 3                        EXITLOOP;
;  797        1945 2                END;
;  798        1946 2
;  799        1947 2 !++
;  800        1948 2 ! THERE WAS NO CORRESPONDING PAL ENTRY.  THEREFORE A NEW ENTRY MUST BE CREATED.
;  801        1949 2 !--
;  802        1950 2 NEW_PTR = PAT$FREEZ((PAL$C_SIZE + A_LONGWORD - 1)/A_LONGWORD);  ! ALLOCATE SPACE FOR NEW ENTRY
;  803        1951 2 IF .TEMP_PTR EQLA 0
;  804        1952 2 THEN
;  805        1953 2         PAT$GL_PAL_LHD = CH$PTR(.NEW_PTR, 0)                    ! SET THE LIST HEAD
;  806        1954 2 ELSE
;  807        1955 2         TEMP_PTR[PAL$L_FLINK] = .NEW_PTR;                       ! LINK IN NEW ENTRY
;  808        1956 2 NEW_PTR[PAL$L_START_ADR] = .START_ADR;                           ! SET STARTING PATCH AREA ADDRESS
;  809        1957 2 NEW_PTR[PAL$L_END_ADR] = .END_ADR;                               ! SET ENDING PATCH AREA ADDRESS
;  810        1958 2 CH$MOVE(A_LONGWORD, PAT_AREA_NAME, NEW_PTR[PAL$L_CS_NAME]);       ! SET PATCH AREA NAME
;  811        1959 2 PAT$GL_SYMTBPTR = .PAT$GL_SYMHEAD;                               ! Use user-defined symbol table
;  812        1960 2 PAT$DEFINE_SYM(PA_NAME_DSC, .NEW_PTR[PAL$L_START_ADR], FALSE);   ! DEFINE SYMBOL AS START OF PATCH AREA
;  813        1961 2 PAT$GL_SYMTBPTR = .TEMP_SYMTB;                            ! Restore label symbol table
;  814        1962 2 PAT_AREA_NAME[3] = .PAT_AREA_NAME[3] + 1;                        ! SET NEW PATCH AREA NAME
;  815        1963 2
;  816        1964 2 !++
;  817        1965 2 ! NOW CHECK THAT THE NEXT PATCH AREA NAME IS BETWEEN "PAA" AND "PAZ".  IF
;  818        1966 2 ! IT IS NOT, THE RESET THE THIRD CHARACTER OF THE NAME TO AN "A" AND
;  819        1967 2 ! INCREMENT THE SECOND LETTER OF THE NAME.  THIS WILL ALLOW THE USER TO DEFINE
;  820        1968 2 ! UP TO 676 PATCH AREAS.
;  821        1969 2 !--
;  822        1970 3 IF .PAT_AREA_NAME[3] GTRU (%ASCII'Z')                             ! CHECK FOR OVERFLOW OF PATCH AREA NAMES
;  823        1971 2 THEN
;  824        1972 3         BEGIN
;  825        1973 3         PAT_AREA_NAME[2] = .PAT_AREA_NAME[2] + 1;                ! INCREMENT THE "A" OF "PAZ"
;  826        1974 3         PAT_AREA_NAME[3] = (%ASCII'A');                          ! CHANGE THE "Z" TO AN "A"
;  827        1975 3         END;
;  828        1976 2
;  829        1977 2 RETURN;
;  830        1978 2
;  831        1979 1 END;                                                             ! END OF PAT$ADD_PAL



                              01FC 00000          .ENTRY  PAT$ADD_PAL, Save R2,R3,R4,R5,R6,R7,R8         ; 1823
          58 00000000G  EF  9E 00002          MOVAB   PAT$DEFINE_SYM, R8                              ;
          57 00000000G  EF  9E 00009          MOVAB   PAT$GL_SYMHEAD, R7                              ;
          56 00000000G  EF  9E 00010          MOVAB   PAT$GL_PAL_LHD, R6                              ;
          55 00000000G  EF  9E 00017          MOVAB   PAT$GL_SYMTBPTR, R5                             ;
          54 00000000'  EF  9E 0001E          MOVAB   PAT_AREA_NAME+3, R4                             ;
          5E                08  C2 00025          SUBL2   #8, SP                                          ;
          53                65  D0 00028          MOVL    PAT$GL_SYMTBPTR, TEMP_SYMTB                     ; 1907
          52                66  D0 0002B          MOVL    PAT$GL_PAL_LHD, TEMP_PTR                        ; 1908
```

```
                                      47  13 0002E       BEQL    4$                                          ;
                  01          0C      AC  D1 00030  1$:  CMPL    PAT_AREA_FLAG, #1                           ; 1920
                                      2A  12 00034       BNEQ    2$                                          ;
            08    A2          08      AC  D0 00036       MOVL    END_ADR, 8(TEMP_PTR)                        ; 1923
                              04      A2  D5 0003B       TSTL    4(TEMP_PTR)                                 ; 1924
                                      76  12 0003E       BNEQ    7$                                          ;
            04    A2          04      AC  D0 00040       MOVL    START_ADR, 4(TEMP_PTR)                      ; 1927
                  6E          FD      A4  9B 00045       MOVZBW  PAT_AREA_NAME, NAME_DESC                    ; 1928
            04    AE          0D      A2  9E 00049       MOVAB   13(R2), NAME_DESC+4                         ; 1929
                  65                  67  D0 0004E       MOVL    PAT$GL_SYMHEAD, PAT$GL_SYMTBPTR             ; 1930
                                      7E  D4 00051       CLRL    -(SP)                                       ; 1931
                              04      AC  DD 00053       PUSHL   START_ADR                                   ;
                              08      AE  9F 00056       PUSHAB  NAME_DESC                                   ;
                  68                  03  FB 00059       CALLS   #3, PAT$DEFINE_SYM                          ;
                  65                  53  D0 0005C       MOVL    TEMP_SYMTB, PAT$GL_SYMTBPTR                 ; 1932
                                      04  0005F          RET                                                 ; 1922
            04    A2          04      AC  D1 00060  2$:  CMPL    START_ADR, 4(TEMP_PTR)                      ; 1936
                                      07  1F 00065       BLSSU   3$                                          ;
            08    A2          08      AC  D1 00067       CMPL    END_ADR, 8(TEMP_PTR)                        ; 1937
                                      48  13 0006C       BEQL    7$                                          ;
                                      62  D5 0006E  3$:  TSTL    (TEMP_PTR)                                  ; 1940
                                      05  13 00070       BEQL    4$                                          ;
                  52                  62  D0 00072       MOVL    (TEMP_PTR), TEMP_PTR                        ; 1942
                                      B9  11 00075       BRB     1$                                          ;
                                      04  DD 00077  4$:  PUSHL   #4                                          ; 1950
     00000000G    EF                  01  FB 00079       CALLS   #1, PAT$FREEZ                               ;
                                      52  D5 00080       TSTL    TEMP_PTR                                    ; 1951
                                      05  12 00082       BNEQ    5$                                          ;
                  66                  50  D0 00084       MOVL    NEW_PTR, PAT$GL_PAL_LHD                     ; 1953
                                      03  11 00087       BRB     6$                                          ;
                  62                  50  D0 00089  5$:  MOVL    NEW_PTR, (TEMP_PTR)                         ; 1955
            04    A0          04      AC  7D 0008C  6$:  MOVQ    START_ADR, 4(NEW_PTR)                       ; 1956
            0C    A0          FD      A4  D0 00091       MOVL    PAT_AREA_NAME, 12(NEW_PTR)                  ; 1958
                  65                  67  D0 00096       MOVL    PAT$GL_SYMHEAD, PAT$GL_SYMTBPTR             ; 1959
                                      7E  D4 00099       CLRL    -(SP)                                       ; 1960
                              04      A0  DD 0009B       PUSHL   4(NEW_PTR)                                  ;
                              01      A4  9F 0009E       PUSHAB  PA_NAME_DSC                                 ;
                  68                  03  FB 000A1       CALLS   #3, PAT$DEFINE_SYM                          ;
                  65                  53  D0 000A4       MOVL    TEMP_SYMTB, PAT$GL_SYMTBPTR                 ; 1961
                                      64  96 000A7       INCB    PAT_AREA_NAME+3                             ; 1962
            5A    8F                  64  91 000A9       CMPB    PAT_AREA_NAME+3, #90                        ; 1970
                                      07  1B 000AD       BLEQU   7$                                          ;
                              FF      A4  96 000AF       INCB    PAT_AREA_NAME+2                             ; 1973
                  64          41      8F  90 000B2       MOVB    #65, PAT_AREA_NAME+3                        ; 1974
                                      04  000B6     7$:  RET                                                 ; 1979

; Routine Size:  183 bytes,    Routine Base:  _PAT$CODE + 0330
```

```
                                                    J 15
PATSPA                                              16-Sep-1984 00:57:14     VAX-11 Bliss-32 V4.0-742                 Page  25
V04-000                                             14-Sep-1984 12:52:47     DISK$VMSMASTER:[PATCH.SRC]PATSPA.B32;1    (7)

;  833          1980  1 END                                                  ! End of module
;  834          1981  0 ELUDOM




                                                              .EXTRN  LIB$SIGNAL

;                                 PSECT SUMMARY

;       Name                    Bytes                     Attributes

;   _PAT$OWN                        12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   _PAT$PLIT                      116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(0)
;   _PAT$CODE                      999  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   .  ABS  .                        0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)




;                         Library Statistics

;                                         -------- Symbols --------        Pages       Processing
;       File                              Total    Loaded   Percent        Mapped      Time

;   _$255$DUA28:[SYSLIB]LIB.L32;1         18619        31         0        1000          00:01.8




;                                 COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LIS$:PATSPA/OBJ=OBJ$:PATSPA MSRC$:PATSPA/UPDATE=(ENH$:PATSPA)

; Size:           999 code + 128 data bytes
; Run Time:          00:34.2
; Elapsed Time:      02:04.8
; Lines/CPU Min:       3479
; Lexemes/CPU-Min:  37166
; Memory Used:   213 pages
; Compilation Complete
```

0303 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY