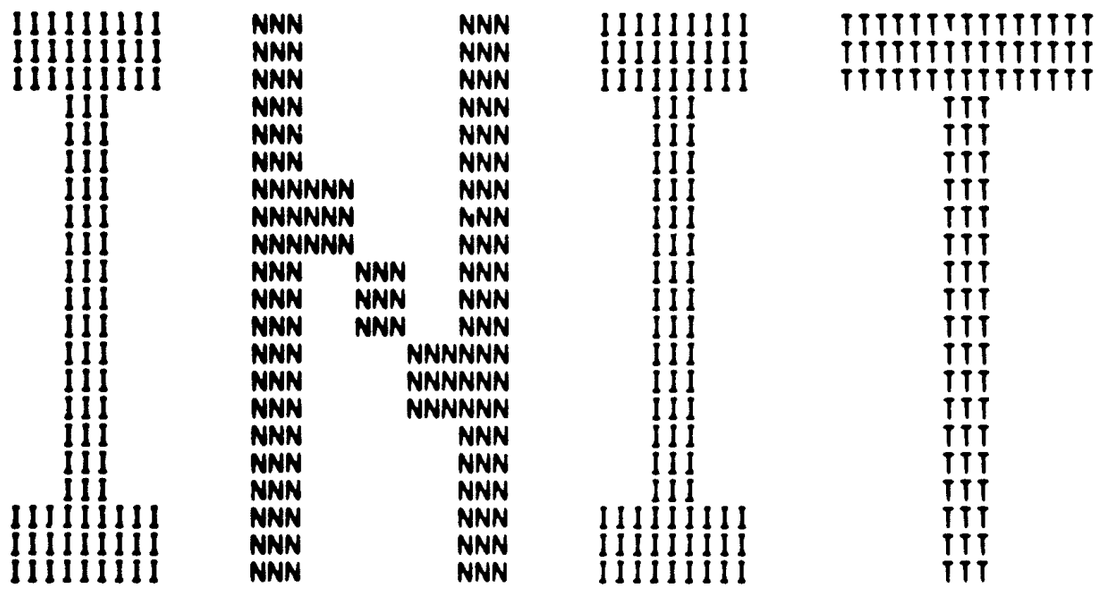
IIIIIIIII NNN NNN IIIIIIIII TTTTTTTTTTTTTTT
IIIIIIIII NNN NNN IIIIIIIII TTTTTTTTTTTTTTT
IIIIIIIII NNN NNN IIIIIIIII TTTTTTTTTTTTTTT
III NNN NNN III TTT
III NNN NNN III TTT
III NNN NNN III TTT
III NNNNNN NNN III TTT
III NNNNNN NNN III TTT
III NNNNNN NNN III TTT
III NNN NNN NNN III TTT
III NNN NNN NNN III TTT
III NNN NNN III TTT
III NNN NNNNNN III TTT
III NNN NNNNNN III TTT
III NNN NNNNNN III TTT
III NNN NNN III TTT
III NNN NNN III TTT
III NNN NNN III TTT
IIIIIIIII NNN NNN IIIIIIIII TTT
IIIIIIIII NNN NNN IIIIIIIII TTT
IIIIIIIII NNN NNN IIIIIIIII TTT

FILEID**ININD1

```
 IIIIII    NN      NN    IIIIII    NN      NN  DDDDDDDD       11
 IIIIII    NN      NN    IIIIII    NN      NN  DDDDDDDD       11
   II      NN      NN      II      NN      NN  DD      DD   1111
   II      NN      NN      II      NN      NN  DD      DD   1111
   II      NNNN    NN      II      NNNN    NN  DD      DD     11
   II      NNNN    NN      II      NNNN    NN  DD      DD     11
   II      NN  NN  NN      II      NN  NN  NN  DD      DD     11
   II      NN  NN  NN      II      NN  NN  NN  DD      DD     11
   II      NN    NNNN      II      NN    NNNN  DD      DD     11
   II      NN    NNNN      II      NN    NNNN  DD      DD     11
   II      NN      NN      II      NN      NN  DD      DD     11          ....
   II      NN      NN      II      NN      NN  DD      DD     11          ....
 IIIIII    NN      NN    IIIIII    NN      NN  DDDDDDDD     111111        ....
 IIIIII    NN      NN    IIIIII    NN      NN  DDDDDDDD     111111        ....

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 ININD1 (
;  2     0002  0                 LANGUAGE (BLISS32),
;  3     0003  0                 IDENT = 'V04-000'
;  4     0004  0                 ) =
;  5     0005  1 BEGIN
;  6     0006  1
;  7     0007  1 !
;  8     0008  1 !**************************************************************************
;  9     0009  1 !*                                                                        *
; 10     0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
; 11     0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
; 12     0012  1 !*  ALL RIGHTS RESERVED.                                                   *
; 13     0013  1 !*                                                                        *
; 14     0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
; 15     0015  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE        *
; 16     0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER    *
; 17     0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
; 18     0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY     *
; 19     0019  1 !*  TRANSFERRED.                                                           *
; 20     0020  1 !*                                                                        *
; 21     0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE   *
; 22     0022  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT       *
; 23     0023  1 !*  CORPORATION.                                                           *
; 24     0024  1 !*                                                                        *
; 25     0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS    *
; 26     0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
; 27     0027  1 !*                                                                        *
; 28     0028  1 !*                                                                        *
; 29     0029  1 !**************************************************************************
; 30     0030  1
; 31     0031  1 !++
; 32     0032  1 !
; 33     0033  1 ! FACILITY:  INIT Utility Structure Level 1
; 34     0034  1 !
; 35     0035  1 ! ABSTRACT:
; 36     0036  1 !
; 37     0037  1 !       This module contains the routines that initialize the contents
; 38     0038  1 !       of a disk's index file for structure level 1: boot and home blocks,
; 39     0039  1 !       bitmap, and the initial file headers.
; 40     0040  1 !
; 41     0041  1 ! ENVIRONMENT:
; 42     0042  1 !
; 43     0043  1 !       STARLET operating system, including privileged system services
; 44     0044  1 !       and internal exec routines.
; 45     0045  1 !
; 46     0046  1 !--
; 47     0047  1 !
; 48     0048  1 !
; 49     0049  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  14-Nov-1977  10:16
; 50     0050  1 !
; 51     0051  1 ! MODIFIED BY:
; 52     0052  1 !
; 53     0053  1 !       V03-004 MCN0140         Maria del C. Nasr       30-Nov-1983
; 54     0054  1 !               Define LABEL_STRING and USER_NAME as bblock descriptors.
; 55     0055  1 !
; 56     0056  1 !       V03-003 ACG0329         Andrew C. Goldstein,    12-Apr-1983  16:30
; 57     0057  1 !               Fold long UIC's into [377,377]
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-002 LMP0021         L. Mark Pilant          5-Apr-1982  15:48
;    60          0060  1 !               Add support for ODS-1 structure version 2.
;    61          0061  1 !
;    62          0062  1 !       V02-004 LMP0001         L. Mark Pilant          4-Nov-1981  16:35
;    63          0063  1 !               Create a multi-header index file if the number of headers
;    64          0064  1 !               to be created cannot be contained in a single header
;    65          0065  1 !
;    66          0066  1 !       V02-003 ACG0191         Andrew C. Goldstein,    25-Feb-1981  18:18
;    67          0067  1 !               Fix size of file header fill
;    68          0068  1 !
;    69          0069  1 !       V02-002 ACG0185         Andrew C. Goldstein,    3-Feb-1981  21:02
;    70          0070  1 !               Add serial number to home block
;    71          0071  1 !
;    72          0072  1 !       V0101   ACG0075         Andrew C. Goldstein,    22-Oct-1979  10:00
;    73          0073  1 !               Remove obsolete characteristics bits
;    74          0074  1 !
;    75          0075  1 !       V0100   ACG00001        Andrew C. Goldstein,  10-Oct-1978  21:27
;    76          0076  1 !       Previous revision history moved to [INIT.SRC]INIT.REV
;    77          0077  1 !**
;    78          0078  1
;    79          0079  1
;    80          0080  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    81          0081  1 REQUIRE 'SRC$:INIDEF.B32';
;    82          0082  1 REQUIRE 'LIBD$:[VMSLIB.OBJ]INITMSG.B32';
;    83          0504  1
;    84          0505  1
;    85          0506  1 FORWARD ROUTINE
;    86          0507  1         INIT_INDEX1     : NOVALUE,      ! main index file initialization
;    87          0508  1         MAKE_POINTER;                   ! construct retrieval pointer
```

```
;    89          0509  1 !+
;    90          0510  1 !
;    91          0511  1 ! Own storage.
;    92          0512  1 !
;    93          0513  1 ! Boot program. The following PDP-11 program will type out the attached
;    94          0514  1 ! message when the volume is booted on a PDP-11, informing the user that
;    95          0515  1 ! this is not a system disk.
;    96          0516  1 !
;    97          0517  1 !-
;    98          0518  1
;    99          0519  1 BIND
;   100          0520  1         BOOT_PROGRAM    = UPLIT WORD (
;   101          0521  1
;   102          0522  1 %O'000240',                     ! BOOTBK: NOP                     ; NOP IDENTIFIES BOOT BLOCK
;   103          0523  1 %O'012706', %O'001000',         !         MOV     #1000,SP        ; SET TEMP STACK
;   104          0524  1 %O'010700',                     !         MOV     PC,R0           ; SET ADDRESS
;   105          0525  1 %O'062700', %O'000036',         !         ADD     #BOTMSG-.,R0    ; OF MESSAGE
;   106          0526  1 %O'112001',                     ! 10$:    MOVB    (R0)+,R1        ; GET NEXT CHARACTER
;   107          0527  1 %O'001403',                     !         BEQ     20$             ; END
;   108          0528  1 %O'004767', %O'000006',         !         CALL    TYPIT           ; NO, PRINT IT
;   109          0529  1 %O'000773',                     !         BR      10$             ; LOOP FOR NEXT CHARACTER
;   110          0530  1 %O'000005',                     ! 20$:    RESET                   ;
;   111          0531  1 %O'000000',                     !         HALT                    ; HALT
;   112          0532  1
;   113          0533  1
;   114          0534  1 %O'110137', %O'177566',         ! TYPIT:  MOVB    R1,@#TPB        ; PRINT CHARACTER
;   115          0535  1 %O'105737', %O'177564',         ! 10$:    TSTB    @#TPS           ; DONE?
;   116          0536  1 %O'100375',                     !         BPL     10$             ; NO, WAIT
;   117          0537  1 %O'000207'                      !         RETURN                  ;
;   118          0538  1
;   119          0539  1
;   120          0540  1                                 ! BOTMSG:
;   121          0541  1
;   122          0542  1                         );
;   123          0543  1
;   124          0544  1 LITERAL
;   125          0545  1         BOOT_PROG_LEN   = 38;
;   126          0546  1
;   127          0547  1 !+
;   128          0548  1 !
;   129          0549  1 ! Boot message. Contains the volume label.
;   130          0550  1 !
;   131          0551  1 !-
;   132          0552  1
;   133          0553  1 BIND
;   134          0554  1         BOOT_MESSAGE    = UPLIT BYTE (13, 10, 10,
;   135          0555  1                                       'is not a system disk', 13, 10, 10, 0);
;   136          0556  1
;   137          0557  1 LITERAL
;   138          0558  1         BOOT_MESG_LEN   = 40;
;   139          0559  1
;   140          0560  1 MACRO
;   141          0561  1         BTB$T_VOLNAME   = 38, 0, 0, 0%; ! volume label in boot block message
;   142          0562  1
;   143          0563  1 !
;   144          0564  1 ! Volume format name string
;   145          0565  1 !
```

```
1
1 BIND
1         FORMAT_NAME     = UPLIT BYTE ('DECFILE11A  ');
1
1 !+
1 !
1 ! Initial file header. The core image file is used since it is the first
1 ! one written.
1 !
1 !-
1
1 BIND
1         INITIAL_HEADER  = UPLIT (
1                                                   ! HEADER area
1         BYTE (FH1$C_LENGTH / 2),                  ! ident area offset
1         BYTE ((FH1$C_LENGTH + FI1$C_LENGTH)/2),   ! map area offset
1         WORD (5, 5),                              ! file ID
1         BYTE (1, 1),                              ! structure version and level
1         WORD (0),                                 ! file owner UIC
1         WORD (0),                                 ! file protection
1         WORD (0),                                 ! file characteristics
1         BYTE (0),                                 ! record type
1         BYTE (0),                                 ! no record attributes
1         WORD (0),                                 ! record size
1         LONG (0, 0),                              ! HIBLK and EFBLK
1         WORD (0),                                 ! EOF byte offset
1         BYTE (0, 0),                              ! bucket size & VFC length
1         WORD (0),                                 ! maximum record length
1         WORD (0),                                 ! default extend size
1         WORD (0, 0, 0, 0, 0, 0),                  ! unused record attributes
1
1                                                   ! IDENT area
1         WORD (%RAD50_11'CORIMG   SYS', 1),        ! file name, type and version
1         WORD (1),                                 ! revision number
1         BYTE (REP 34 OF (0)),                     ! dates
1
1                                                   ! MAP area
1         BYTE (0),                                 ! file segment number
1         BYTE (0),                                 ! extension RVN
1         WORD (0, 0),                              ! extension file ID
1         BYTE (1, 3),                              ! map pointer count & LBN size
1         BYTE (0),                                 ! map words in use
1         BYTE ((512-2-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH)/2)
1                                                   ! map words available
1
1         );
```

```
;  193          0612  1 GLOBAL ROUTINE INIT_INDEX1 : NOVALUE =
;  194          0613  1
;  195          0614  1 !++
;  196          0615  1 !
;  197          0616  1 ! FUNCTIONAL DESCRIPTION:
;  198          0617  1 !
;  199          0618  1 !       This routine initializes the contents of the disk's index file.
;  200          0619  1 !       It writes a dummy boot block, the home blocks, index file bitmap,
;  201          0620  1 !       and the initial headers.
;  202          0621  1 !
;  203          0622  1 !
;  204          0623  1 ! CALLING SEQUENCE:
;  205          0624  1 !       INIT_INDEX1 ()
;  206          0625  1 !
;  207          0626  1 ! INPUT PARAMETERS:
;  208          0627  1 !       NONE
;  209          0628  1 !
;  210          0629  1 ! IMPLICIT INPUTS:
;  211          0630  1 !       parser data base
;  212          0631  1 !       allocation table in INIDSK
;  213          0632  1 !
;  214          0633  1 ! OUTPUT PARAMETERS:
;  215          0634  1 !       NONE
;  216          0635  1 !
;  217          0636  1 ! IMPLICIT OUTPUTS:
;  218          0637  1 !       NONE
;  219          0638  1 !
;  220          0639  1 ! ROUTINE VALUE:
;  221          0640  1 !       NONE
;  222          0641  1 !
;  223          0642  1 ! SIDE EFFECTS:
;  224          0643  1 !       index file blocks written
;  225          0644  1 !
;  226          0645  1 !--
;  227          0646  1
;  228          0647  2 BEGIN
;  229          0648  2
;  230          0649  2 BUILTIN
;  231          0650  2         ROT;
;  232          0651  2
;  233          0652  2 LOCAL
;  234          0653  2         DATE_TIME       : VECTOR [13, BYTE], ! buffer for current date/time
;  235          0654  2         LBN,                                 ! current LBN
;  236          0655  2         MAP_COUNT,                           ! count field of map pointer
;  237          0656  2         MAP_LBN,                             ! start LBN of current map pointer
;  238          0657  2         IDXFILE_EXT_CNT,                     ! remaining count for index file size
;  239          0658  2         IDXFILE_EXT_LBN,                     ! starting block for remaining count
;  240          0659  2         EXTENSION_FID,                       ! file number for extension headers
;  241          0660  2         MAP_FULL_STATUS;                     ! map area full indicator
;  242          0661  2
;  243          0662  2 LITERAL
;  244          0663  2         IDXFILE_EXT_FID = 6;                 ! first index file extension header file number
;  245          0664  2
;  246          0665  2 EXTERNAL
;  247          0666  2         INIT_OPTIONS    : BITVECTOR,         ! command options
;  248          0667  2         BUFFER          : BBLOCK,            ! I/O buffer
;  249          0668  2         VOLUME_SIZE,                         ! size of volume rounded to next cluster
```

```
;  250          0669  2          PROTECTION,                     ! volume protection
;  251          0670  2          FILE_PROT,                      ! default file protection
;  252          0671  2          MAXIMUM,                        ! maximum number of files on volume
;  253          0672  2          OWNER_UIC,                      ! volume owner
;  254          0673  2          EXTENSION,                      ! default file extend
;  255          0674  2          WINDOW,                         ! default window size
;  256          0675  2          ACCESSED,                       ! default directory LRU limit
;  257          0676  2          SERIAL_NUMBER,                  ! pack serial number
;  258          0677  2          BADBLOCK_TOTAL,                 ! count of bad blocks on volume
;  259          0678  2          ALLOC_TABLE_CNT : VECTOR,       ! allocation count table
;  260          0679  2          ALLOC_TABLE_LBN : VECTOR,       ! allocation LBN table
;  261          0680  2          BADBLOCK_CNT    : VECTOR,       ! bad block count table
;  262          0681  2          BADBLOCK_LBN    : VECTOR,       ! bad LBN table
;  263          0682  2          BOOTBLOCK_CNT,                  ! boot block cluster block count
;  264          0683  2          BOOTBLOCK_LBN,                  ! boot block cluster LBN
;  265          0684  2          HOMEBLOCK1_CNT,                 ! home block 1 cluster block count
;  266          0685  2          HOMEBLOCK1_LBN,                 ! home block 1 cluster LBN
;  267          0686  2          IDXFILE_CNT,                    ! initial index file count
;  268          0687  2          IDXFILE_LBN,                    ! initial index file LBN
;  269          0688  2          BITMAP_CNT,                     ! storage bitmap block count
;  270          0689  2          BITMAP_LBN,                     ! storage bitmap LBN
;  271          0690  2          MFD_CNT,                        ! MFD block count
;  272          0691  2          MFD_LBN,                        ! MFD LBN
;  273          0692  2          LABEL_STRING    : BBLOCK [DSC$C_S_BLN], ! string descriptor of volume label
;  274          0693  2          USER_NAME       : BBLOCK [DSC$C_S_BLN]; ! string descriptor of user name
;  275          0694  2
;  276          0695  2 EXTERNAL LITERAL
;  277          0696  2          BOOTBLOCK_IDX   : UNSIGNED (6), ! allocation table boot block index
;  278          0697  2          IDXFILE_IDX     : UNSIGNED (6); ! allocation table index file index
;  279          0698  2
;  280          0699  2 BIND
;  281          0700  2          IDENT_AREA      = BUFFER + FH1$C_LENGTH : BBLOCK,
;  282          0701  2          MAP_AREA        = BUFFER + FH1$C_LENGTH + FI1$C_LENGTH : BBLOCK;
;  283          0702  2
;  284          0703  2 EXTERNAL ROUTINE
;  285          0704  2          GET_TIME,                       ! get ASCII date/time string
;  286          0705  2          CHECKSUM2,                      ! compute block checksum
;  287          0706  2          READ_BLOCK,                     ! read a block from the disk
;  288          0707  2          WRITE_BLOCK;                    ! write block to disk
;  289          0708  2
;  290          0709  2
;  291          0710  2 ! First block to write is the boot block. Set up the message routine for
;  292          0711  2 ! the -11 and build the message.
;  293          0712  2 !
;  294          0713  2
;  295          0714  2 CH$COPY (BOOT_PROG_LEN, BOOT_PROGRAM,
;  296          0715  2           BOOT_MESG_LEN, BOOT_MESSAGE,
;  297          0716  2           0, %T2, BUFFER);
;  298          0717  2 CH$MOVE (.LABEL_STRING [DSC$W_LENGTH],
;  299          0718  2           .LABEL_STRING [DSC$A_POINTER],
;  300          0719  2           BUFFER[BTB$T_VOLNAME]);
;  301          0720  2
;  302          0721  2 WRITE_BLOCK (.BOOTBLOCK_LBN, BUFFER);
;  303          0722  2
;  304          0723  2 ! Fold the file and volume owner UIC into 16 bits. If the specified value
;  305          0724  2 ! is outside the 16 bit range, set it to [377,377].
;  306          0725  2 !
```

```
;   307          0726  2
;   308          0727  2 IF .OWNER_UIC<8,8> NEQ 0
;   309          0728  2 OR .OWNER_UIC<24,8> NEQ 0
;   310          0729  2 THEN
;   311          0730  3     BEGIN
;   312          0731  3     OWNER_UIC<0,8> = -1;
;   313          0732  3     OWNER_UIC<8,8> = 0;
;   314          0733  3     OWNER_UIC<16,8> = -1;
;   315          0734  3     OWNER_UIC<24,8> = 0;
;   316          0735  2     END;
;   317          0736  2
;   318          0737  2 ! Now construct and write the home block.
;   319          0738  2 !
;   320          0739  2
;   321          0740  2 GET_TIME (DATE_TIME[0]);
;   322          0741  2 CH$FILL (0, 512, BUFFER);
;   323          0742  2
;   324          0743  2 BUFFER[HM1$W_IBMAPSIZE] = (.MAXIMUM + 4095) / 4096;
;   325          0744  2 BUFFER[HM1$L_IBMAPLBN]  = ROT (.IDXFILE_LBN, 16);
;   326          0745  2 BUFFER[HM1$W_MAXFILES]  = .MAXIMUM;
;   327          0746  2 BUFFER[HM1$W_CLUSTER]   = 1;
;   328          0747  2 BUFFER[HM1$W_STRUCLEV]  = HM1$C_LEVEL1;
;   329          0748  2 CH$COPY (.LABEL_STRING [DSC$W_LENGTH], .LABEL_STRING [DSC$A_POINTER],
;   330          0749  2           0, HM1$S_VOLNAME, BUFFER[HM1$T_VOLNAME]);
;   331          0750  2 (BUFFER[HM1$W_VOLOWNER])<0,8>   = .OWNER_UIC<0,8>;
;   332          0751  2 (BUFFER[HM1$W_VOLOWNER])<8,8>   = .OWNER_UIC<16,8>;
;   333          0752  2 BUFFER[HM1$W_PROTECT]   = .PROTECTION;
;   334          0753  2 BUFFER[HM1$W_FILEPROT]  = .FILE_PROT;
;   335          0754  2 BUFFER[HM1$B_WINDOW]    = .WINDOW;
;   336          0755  2 BUFFER[HM1$B_EXTEND]    = .EXTENSION;
;   337          0756  2 BUFFER[HM1$B_LRU_LIM]   = .ACCESSED;
;   338          0757  2 BUFFER[HM1$L_SERIALNUM] = .SERIAL_NUMBER;
;   339          0758  2
;   340          0759  2 CH$MOVE (13, DATE_TIME[0], BUFFER[HM1$T_CREDATE]);
;   341          0760  2 CH$COPY (.LABEL_STRING [DSC$W_LENGTH], .LABEL_STRING [DSC$A_POINTER],
;   342          0761  2           32, HM1$S_VOLNAME2, BUFFER[HM1$T_VOLNAME2]);
;   343          0762  2 CH$COPY (.USER_NAME [DSC$W_LENGTH], .USER_NAME [DSC$A_POINTER],
;   344          0763  2           32, HM1$S_OWNERNAME, BUFFER[HM1$T_OWNERNAME]);
;   345          0764  2 CH$MOVE (HM1$S_FORMAT, FORMAT_NAME, BUFFER[HM1$T_FORMAT]);
;   346          0765  2
;   347          0766  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (HM1$W_CHECKSUM1));
;   348          0767  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (HM1$W_CHECKSUM2));
;   349          0768  2 WRITE_BLOCK (.HOMEBLOCK1_LBN, BUFFER);
;   350          0769  2
;   351          0770  2 ! Now write out the initial index file bitmap. The first block contains the
;   352          0771  2 ! reserved files marked in use; the rest are all zero.
;   353          0772  2 !
;   354          0773  2
;   355          0774  2 CH$FILL (0, 512, BUFFER);
;   356          0775  2 BUFFER<0,32> = %B'11111';
;   357          0776  2 LBN = .IDXFILE_LBN;
;   358          0777  2 WRITE_BLOCK (.LBN, BUFFER);
;   359          0778  2
;   360          0779  2 BUFFER<0,32> = 0;
;   361          0780  2 DECR J FROM (.MAXIMUM+4095)/4096-1 TO 1 DO
;   362          0781  3     BEGIN
;   363          0782  3     LBN = .LBN + 1;
```

```
;  364          0783  3     WRITE_BLOCK (.LBN, BUFFER);
;  365          0784  2     END;
;  366          0785  2
;  367          0786  2 ! Construct and write the initial core image file header.
;  368          0787  2 !
;  369          0788  2
;  370          0789  2 CH$COPY (FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH, INITIAL_HEADER,
;  371          0790  2           0, 512, BUFFER);
;  372          0791  2 (BUFFER[FH1$W_FILEOWNER])<0,8> = .OWNER_UIC<0,8>;
;  373          0792  2 (BUFFER[FH1$W_FILEOWNER])<8,8> = .OWNER_UIC<16,8>;
;  374          0793  2 BUFFER[FH1$W_FILEPROT] = .FILE_PROT;
;  375          0794  2 CH$MOVE (13, DATE_TIME, IDENT_AREA[FI1$T_REVDATE]);
;  376          0795  2 CH$MOVE (13, DATE_TIME, IDENT_AREA[FI1$T_CREDATE]);
;  377          0796  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  378          0797  2 WRITE_BLOCK (.LBN + 5, BUFFER);
;  379          0798  2
;  380          0799  2 ! Turn the header into the index file header and write it.
;  381          0800  2 !
;  382          0801  2
;  383          0802  2 BUFFER[FH1$W_FID_NUM] = 1;
;  384          0803  2 BUFFER[FH1$W_FID_SEQ] = 1;
;  385          0804  2 (IDENT_AREA[FI1$Q_FILENAME])<0,32> = %RAD50_11 'INDEXF';
;  386          0805  2 INCR J FROM BOOTBLOCK_IDX TO IDXFILE_IDX-1 DO
;  387          0806  3     BEGIN
;  388          0807  3     IF .ALLOC_TABLE_CNT[.J] NEQ 0
;  389          0808  3     THEN MAKE_POINTER (ALLOC_TABLE_CNT[.J], ALLOC_TABLE_LBN[.J], 0);
;  390          0809  2     END;
;  391          0810  2
;  392          0811  2 ! Now that the basic information has been accounted for, account for
;  393          0812  2 ! space required to keep the index file headers.  This gets interresting
;  394          0813  2 ! if it is necessary to generate extension headers.
;  395          0814  2 !
;  396          0815  2
;  397          0816  2 IDXFILE_EXT_CNT = .ALLOC_TABLE_CNT[IDXFILE_IDX];
;  398          0817  2 IDXFILE_EXT_LBN = .ALLOC_TABLE_LBN[IDXFILE_IDX];
;  399          0818  2
;  400          0819  2 IF NOT MAKE_POINTER (IDXFILE_EXT_CNT, IDXFILE_EXT_LBN, 1)
;  401          0820  2 THEN
;  402          0821  3     BEGIN
;  403          0822  3
;  404          0823  3     LOCAL       BUFFER1         : BBLOCK [512]; ! temp home block storage
;  405          0824  3
;  406          0825  3     READ_BLOCK (.HOMEBLOCK1_LBN, BUFFER1);
;  407          0826  3     BUFFER1[HM1$W_STRUCLEV] = HM1$C_LEVEL2;
;  408          0827  3     CHECKSUM2 (BUFFER1, $BYTEOFFSET (HM1$W_CHECKSUM1));
;  409          0828  3     CHECKSUM2 (BUFFER1, $BYTEOFFSET (HM1$W_CHECKSUM2));
;  410          0829  3     WRITE_BLOCK (.HOMEBLOCK1_LBN, BUFFER1);
;  411          0830  3
;  412          0831  3     EXTENSION_FID = IDXFILE_EXT_FID;
;  413          0832  3     MAP_AREA[FM1$B_EX_RVN] = 0;
;  414          0833  3     MAP_AREA[FM1$W_EX_FILNUM] = .EXTENSION_FID;
;  415          0834  3     MAP_AREA[FM1$W_EX_FILSEQ] = .EXTENSION_FID;
;  416          0835  3     MAP_AREA[FM1$B_EX_SEGNUM] = 0;
;  417          0836  3     CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  418          0837  3     WRITE_BLOCK(.LBN + 1, BUFFER);
;  419          0838  3
;  420          0839  3     DO
```

```
;  421          0840  4          BEGIN
;  422          0841  4          BUFFER[FH1$W_FID_NUM] = .EXTENSION_FID;
;  423          0842  4          BUFFER[FH1$W_FID_SEQ] = .EXTENSION_FID;
;  424          0843  4          MAP_AREA[FM1$B_INUSE] = 0;
;  425          0844  4          CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGT
;  426          0845  4          MAP_FULL_STATUS = MAKE_POINTER (IDXFILE_EXT_CNT, IDXFILE_EXT_LBN, 1);
;  427          0846  4          MAP_AREA[FM1$W_EX_FILNUM] = .EXTENSION_FID + 1;
;  428          0847  4          MAP_AREA[FM1$W_EX_FILSEQ] = .EXTENSION_FID + 1;
;  429          0848  4          IF .MAP_FULL_STATUS
;  430          0849  4          THEN
;  431          0850  5              BEGIN
;  432          0851  5              MAP_AREA[FM1$W_EX_FILNUM] = 0;
;  433          0852  5              MAP_AREA[FM1$W_EX_FILSEQ] = 0;
;  434          0853  4              END;
;  435          0854  4          MAP_AREA[FM1$B_EX_SEGNUM] = .MAP_AREA[FM1$B_EX_SEGNUM] + 1;
;  436          0855  4          CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  437          0856  4          WRITE_BLOCK (.LBN + .EXTENSION_FID, BUFFER);
;  438          0857  4          EXTENSION_FID = .EXTENSION_FID + 1;
;  439          0858  4          END
;  440          0859  3      UNTIL .MAP_FULL_STATUS NEQ 0;
;  441          0860  3
;  442          0861  3  ! Mark any created index file extension headers as in use
;  443          0862  3  !
;  444          0863  3
;  445          0864  3      READ_BLOCK (.IDXFILE_LBN, BUFFER);
;  446          0865  3      BUFFER<IDXFILE_EXT_FID-1,.EXTENSION_FID - IDXFILE_EXT_FID> = %X'FFFFFFFF';
;  447          0866  3      WRITE_BLOCK (.IDXFILE_LBN, BUFFER);
;  448          0867  3
;  449          0868  3  ! Retrieve the primary index file header
;  450          0869  3  !
;  451          0870  3      READ_BLOCK (.LBN + 1, BUFFER);
;  452          0871  3      MAP_AREA[FM1$W_EX_FILNUM] = 0;
;  453          0872  3      MAP_AREA[FM1$W_EX_FILSEQ] = 0;
;  454          0873  3      END
;  455          0874  2  ELSE
;  456          0875  3      BEGIN
;  457          0876  3      CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  458          0877  3      WRITE_BLOCK (.LBN + 1, BUFFER);
;  459          0878  2      END;
;  460          0879  2
;  461          0880  2  ! Turn the file header into the bad block file header and write it.
;  462          0881  2  !
;  463          0882  2
;  464          0883  2  CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
;  465          0884  2  BUFFER[FH1$W_FID_NUM] = 3;
;  466          0885  2  BUFFER[FH1$W_FID_SEQ] = 3;
;  467          0886  2
;  468          0887  2  MAP_AREA[FM1$B_INUSE] = 0;
;  469          0888  2  MAP_COUNT = 0;
;  470          0889  2  INCR J FROM 0 TO .BADBLOCK_TOTAL-1 DO
;  471          0890  2      MAP_COUNT = .MAP_COUNT + .BADBLOCK_CNT[.J];
;  472          0891  2
;  473          0892  2  (IDENT_AREA[FI1$W_FILENAME])<0,32> = %RAD50_11 'BADBLK';
;  474          0893  2  INCR J FROM 0 TO .BADBLOCK_TOTAL-1 DO
;  475          0894  3      BEGIN
;  476          0895  3      IF .MAP_AREA[FM1$B_INUSE] GTR (512 - FH1$C_LENGTH - FI1$C_LENGTH - FM1$C_LENGTH - 4 - 2) / 2
;  477          0896  3      THEN ERR_EXIT (INIT$_MAXBAD);
```

```
;  478          0897  3     MAKE_POINTER (BADBLOCK_CNT[.J], BADBLOCK_LBN[.J], 0);
;  479          0898  2     END;
;  480          0899  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  481          0900  2 WRITE_BLOCK (.LBN + 3, BUFFER);
;  482          0901  2
;  483          0902  2 ! Turn the file header into the storage map file header and write it.
;  484          0903  2 !
;  485          0904  2
;  486          0905  2 CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
;  487          0906  2 BUFFER[FH1$W_FID_NUM] = 2;
;  488          0907  2 BUFFER[FH1$W_FID_SEQ] = 2;
;  489          0908  2 MAP_AREA[FM1$B_INUSE] = 0;
;  490          0909  2
;  491          0910  2 (IDENT_AREA[FI1$W_FILENAME])<0,32> = %RAD50_11 'BITMAP';
;  492          0911  2 MAKE_POINTER (%REF(1), BITMAP_LBN, 0);
;  493          0912  2 MAKE_POINTER (%REF(.BITMAP_CNT-1), %REF(.BITMAP_LBN+1), 0);
;  494          0913  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  495          0914  2 WRITE_BLOCK (.LBN + 2, BUFFER);
;  496          0915  2
;  497          0916  2 ! Turn the file header into the MFD header and write it.
;  498          0917  2 !
;  499          0918  2
;  500          0919  2 CH$FILL (0, 512-FH1$C_LENGTH-FI1$C_LENGTH-FM1$C_LENGTH, BUFFER+FH1$C_LENGTH+FI1$C_LENGTH+FM1$C_LENGTH);
;  501          0920  2 MAP_AREA[FM1$B_INUSE] = 0;
;  502          0921  2 BUFFER[FH1$W_FID_NUM] = 4;
;  503          0922  2 BUFFER[FH1$W_FID_SEQ] = 4;
;  504          0923  2 BUFFER[FH1$V_CONTIG] = 1;
;  505          0924  2 BBLOCK [BUFFER[FH1$W_RECATTR], FAT$B_RTYPE] = FAT$C_FIXED;
;  506          0925  2 BBLOCK [BUFFER[FH1$W_RECATTR], FAT$W_RSIZE] = 16;
;  507          0926  2 BBLOCK [BUFFER[FH1$W_RECATTR], FAT$L_EFBLK] = ROT (2, 16);
;  508          0927  2 BBLOCK [BUFFER[FH1$W_RECATTR], FAT$L_HIBLK] = ROT (.MFD_CNT, 16);
;  509          0928  2
;  510          0929  2 (IDENT_AREA[FI1$W_FILENAME])<0,32> = %RAD50_11 '000000';
;  511          0930  2 (IDENT_AREA[FI1$W_FILENAME])<32,32> = %RAD50_11 '   DIR';
;  512          0931  2 MAKE_POINTER (MFD_CNT, MFD_LBN, 0);
;  513          0932  2 CHECKSUM2 (BUFFER, $BYTEOFFSET (FH1$W_CHECKSUM));
;  514          0933  2 WRITE_BLOCK (.LBN + 4, BUFFER);
;  515          0934  2
;  516          0935  1 END;                                    ! end of routine INIT_INDEX


                                                                .TITLE  ININD1
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

0006  09F7  0303  9401  001E  65C0  11C0  0200  15C6  00A0  00000 P.AAA:  .WORD   160, 5574, 512, 4544, 26048, 30, -27647, -  ;
      0087  80FD  FF74  8BDF  FF76  905F  0000  0005  01FB  00014                 771, 2551, 6, 507, 5, 0, -28577, -138, -  ;
                                                                                  -29729, -140, -32515, 135
                                                      0A  0A  0D  00026 P.AAB:  .BYTE   13, 10, 10                        ;
73  69  20  20  20  20  20  20  20  20  20  20  20  20  20  0D  00029         .ASCII  \              is not a system disk\  ;
64  20  6D  65  74  73  79  73  20  61  20  74  6F  6E  20  00038
                                                      6B  73  69  00047
                                                  00  0A  0A  0D  0004A         .BYTE   13, 10, 10, 0                      ;
            20  20  41  31  31  45  4C  49  46  43  45  44  0004E P.AAC:  .ASCII  \DECFILE11A  \                          ;
                                                                0005A         .BLKB   2
                                                          17  0005C P.AAD:  .BYTE   23                                  ;
```

```
                                    2E  0005D          .BYTE   46                                                      ;
                              0005  0005  0005E          .WORD   5, 5                                                    ;
                                 01  01  00062          .BYTE   1, 1                                                    ;
                                   0000  00064          .WORD   0                                                       ;
                                   0000  00066          .WORD   0                                                       ;
                                   0000  00068          .WORD   0                                                       ;
                                     00  0006A          .BYTE   0                                                       ;
                                     00  0006B          .BYTE   0                                                       ;
                                   0000  0006C          .WORD   0                                                       ;
                         00000000  00000000  0006E          .LONG   0, 0                                                    ;
                                   0000  00076          .WORD   0                                                       ;
                                00  00  00078          .BYTE   0, 0                                                    ;
                                   0000  0007A          .WORD   0                                                       ;
                                   0000  0007C          .WORD   0                                                       ;
0000  0000  0000  0000  0000  0000  0007E          .WORD   0, 0, 0, 0, 0, 0                                        ;
   7A  BB  00  00  3A  4F  15  2A  0008A          .RAD50  \CORIMG   SYS\                                          ;
                                   0001  00092          .WORD   1                                                       ;
                                   0001  00094          .WORD   1                                                       ;
                                     00#  00096          .BYTE   0[34]                                                   ;
                                     00  000B8          .BYTE   0                                                       ;
                                     00  000B9          .BYTE   0                                                       ;
                              0000  0000  000BA          .WORD   0, 0                                                    ;
                                 03  01  000BE          .BYTE   1, 3                                                    ;
                                     00  000C0          .BYTE   0                                                       ;
                                     CC  000C1          .BYTE   -52                                                     ;

                                                        BOOT_PROGRAM=       P.AAA
                                                        BOOT_MESSAGE=       P.AAB
                                                        FORMAT_NAME=        P.AAC
                                                        INITIAL_HEADER=     P.AAD
                                                               .EXTRN   INIT_OPTIONS, BUFFER
                                                               .EXTRN   VOLUME_SIZE, PROTECTION
                                                               .EXTRN   FILE_PROT, MAXIMUM
                                                               .EXTRN   OWNER_UIC, EXTENSION
                                                               .EXTRN   WINDOW, ACCESSED
                                                               .EXTRN   SERIAL_NUMBER, BADBLOCK_TOTAL
                                                               .EXTRN   ALLOC_TABLE_CNT
                                                               .EXTRN   ALLOC_TABLE_LBN
                                                               .EXTRN   BADBLOCK_CNT, BADBLOCK_LBN
                                                               .EXTRN   BOOTBLOCK_CNT, BOOTBLOCK_LBN
                                                               .EXTRN   HOMEBLOCKT_CNT, HOMEBLOCK1_LBN
                                                               .EXTRN   IDXFILE_CNT, IDXFILE_LBN
                                                               .EXTRN   BITMAP_CNT, BITMAP_LBN
                                                               .EXTRN   MFD_CNT, MFD_LBN
                                                               .EXTRN   LABEL_STRING, USER_NAME
                                                               .EXTRN   BOOTBLOCK_IDX, IDXFILE_IDX
                                                               .EXTRN   GET_TIME, CHECKSUM2
                                                               .EXTRN   READ_BLOCK, WRITE_BLOCK

                                                               .PSECT   $CODE$,NOWRT,2

                                   0FFC 00000          .ENTRY   INIT_INDEX1, Save R2,R3,R4,R5,R6,R7,R8,R9,- ; 0612
                                                                R10,R11                                        ;
                         5B      0000G  CF  9E 00002          MOVAB   WRITE_BLOCK, R11                               ;
                         5A      0000G  CF  9E 00007          MOVAB   BUFFER, R10                                    ;
                         5E      FDE0   CE  9E 0000C          MOVAB   -544(SP), SP                                   ;
   6A      0000'  CF              26  28 00011          MOVC3   #38, BOOT_PROGRAM, BUFFER                      ; 0714
```

```
01DA  8F          00  0000'  CF                28  2C  00017        MOVC5   #40, BOOT_MESSAGE, #0, #474, (R3)           ;
                                               63      00020                                                              ;
            26    AA  0000G  DF         0000G  CF  28  00021        MOVC3   LABEL_STRING, @LABEL_STRING+4, BUFFER+38    ; 0719
                                               5A  DD  0002A        PUSHL   R10                                         ; 0721
                                        0000G  CF  DD  0002C        PUSHL   BOOTBLOCK_LBN                               ;
                             6B                02  FB  00030        CALLS   #2, WRITE_BLOCK                             ;
                                        0000G  CF  95  00033        TSTB    OWNER_UIC+1                                 ; 0727
                                               06  12  00037        BNEQ    1$                                          ;
                                        0000G  CF  95  00039        TSTB    OWNER_UIC+3                                 ; 0728
                                               09  13  0003D        BEQL    2$                                          ;
                      0000G  CF  00FF00FF      8F  D0  0003F  1$:   MOVL    #16711935, OWNER_UIC                        ; 0731
                                       F0      AD  9F  00048  2$:   PUSHAB  DATE_TIME                                   ; 0740
                      0000G  CF                01  FB  0004B        CALLS   #1, GET_TIME                                ;
0200  8F          00         6E                00  2C  00050        MOVC5   #0, (SP), #0, #512, BUFFER                  ; 0741
                                               6A      00057                                                              ;
                  50  0000G  CF  00000FFF      8F  C1  00058        ADDL3   #4095, MAXIMUM, R0                          ; 0743
                  51         50  00001000      8F  C7  00062        DIVL3   #4096, R0, R1                               ;
                             6A                51  B0  0006A        MOVW    R1, BUFFER                                  ;
            02    AA  0000G  CF                10  9C  0006D        ROTL    #16, IDXFILE_LBN, BUFFER+2                  ; 0744
                         06  AA         0000G  CF  B0  00074        MOVW    MAXIMUM, BUFFER+6                           ; 0745
                         08  AA                01  B0  0007A        MOVW    #1, BUFFER+8                                ; 0746
                         0C  AA         0101   8F  B0  0007E        MOVW    #257, BUFFER+12                             ; 0747
      0C          00  0000G  DF         0000G  CF  2C  00084        MOVC5   LABEL_STRING, @LABEL_STRING+4, #0, #12, -   ; 0749
                                           0E  AA      0008D                BUFFER+14                                   ;
                         1E  AA         0000G  CF  90  0008F        MOVB    OWNER_UIC, BUFFER+30                        ; 0750
                         1F  AA         0000G  CF  90  00095        MOVB    OWNER_UIC+2, BUFFER+31                      ; 0751
                         20  AA         0000G  CF  B0  0009B        MOVW    PROTECTION, BUFFER+32                       ; 0752
                         24  AA         0000G  CF  B0  000A1        MOVW    FILE_PROT, BUFFER+36                        ; 0753
                         2C  AA         0000G  CF  90  000A7        MOVB    WINDOW, BUFFER+44                           ; 0754
                         2D  AA         0000G  CF  90  000AD        MOVB    EXTENSION, BUFFER+45                        ; 0755
                         2E  AA         0000G  CF  90  000B3        MOVB    ACCESSED, BUFFER+46                         ; 0756
                       01C8  CA         0000G  CF  D0  000B9        MOVL    SERIAL_NUMBER, BUFFER+456                   ; 0757
            3C    AA     F0  AD                0D  28  000C0        MOVC3   #13, DATE_TIME, BUFFER+60                   ; 0759
      0C          20  0000G  DF         0000G  CF  2C  000C6        MOVC5   LABEL_STRING, @LABEL_STRING+4, #32, #12, -  ; 0761
                                        01D8   CA      000CF                BUFFER+472                                  ;
      0C          20  0000G  DF         0000G  CF  2C  000D2        MOVC5   USER_NAME, @USER_NAME+4, #32, #12, -        ; 0763
                                        01E4   CA      000DB                BUFFER+484                                  ;
          01F0    CA  0000'  CF                0C  28  000DE        MOVC3   #12, FORMAT_NAME, BUFFER+496                ; 0764
                                               3A  DD  000E6        PUSHL   #58                                         ; 0766
                                               5A  DD  000E8        PUSHL   R10                                         ;
                      0000G  CF                02  FB  000EA        CALLS   #2, CHECKSUM2                               ;
                             7E         01FE   8F  3C  000EF        MOVZWL  #510, -(SP)                                 ; 0767
                                               5A  DD  000F4        PUSHL   R10                                         ;
                      0000G  CF                02  FB  000F6        CALLS   #2, CHECKSUM2                               ;
                                               5A  DD  000FB        PUSHL   R10                                         ; 0768
                                        0000G  CF  DD  000FD        PUSHL   HOMEBLOCK1_LBN                              ;
                             6B                02  FB  00101        CALLS   #2, WRITE_BLOCK                             ;
0200  8F          00         6E                00  2C  00104        MOVC5   #0, (SP), #0, #512, BUFFER                  ; 0774
                                               6A      0010B                                                              ;
                             6A                1F  D0  0010C        MOVL    #31, BUFFER                                 ; 0775
                             57         0000G  CF  D0  0010F        MOVL    IDXFILE_LBN, LBN                            ; 0776
                                        0480   8F  BB  00114        PUSHR   #^M<R7,R10>                                 ; 0777
                             6B                02  FB  00118        CALLS   #2, WRITE_BLOCK                             ;
                                               6A  D4  0011B        CLRL    BUFFER                                      ; 0779
                  52  0000G  CF  00000FFF      8F  C1  0011D        ADDL3   #4095, MAXIMUM, R2                          ; 0780
                             52  00001000      8F  C6  00127        DIVL2   #4096, R2                                   ;
                                               09  11  0012E        BRB     4$                                          ;
```

```
                                                57  D6 00130 3$:    INCL    LBN                                             ; 0782
                                          0480  8F  BB 00132        PUSHR   #^M<R7,R10>                                     ; 0783
                                     6B         02  FB 00136        CALLS   #2, WRITE_BLOCK
                                     F4         52  F5 00139 4$:    SOBGTR  J, 3$                                           ; 0780
    0200    8F        00     0000'  CF  0066   8F  2C 0013C        MOVC5   #102, INITIAL_HEADER, #0, #512, BUFFER          ; 0789
                                                6A      00147
                                08   AA 0000G  CF  90 00148        MOVB    OWNER_UIC, BUFFER+8                             ; 0791
                                09   AA 0000G  CF  90 0014E        MOVB    OWNER_UIC+2, BUFFER+9                           ; 0792
                                0A   AA 0000G  CF  B0 00154        MOVW    FILE_PROT, BUFFER+10                            ; 0793
                          3A    AA   F0  AD         0D  28 0015A  MOVC3   #13, DATE_TIME, IDENT_AREA+12                   ; 0794
                          47    AA   F0  AD         0D  28 00160  MOVC3   #13, DATE_TIME, IDENT_AREA+25                   ; 0795
                                7E   01FE 8F  3C 00166        MOVZWL  #510, -(SP)                                     ; 0796
                                                5A  DD 0016B        PUSHL   R10
                                0000G  CF           02  FB 0016D  CALLS   #2, CHECKSUM2
                                                5A  DD 00172        PUSHL   R10                                             ; 0797
                                          05    A7  9F 00174        PUSHAB  5(LBN)
                                     6B         02  FB 00177        CALLS   #2, WRITE_BLOCK
                                02   AA 00010001 8F  D0 0017A        MOVL    #65537, BUFFER+2                                ; 0802
                                2E   AA 23063A74 8F  D0 00182        MOVL    #587610740, IDENT_AREA                          ; 0804
                          52         00G         01  C3 0018A        SUBL3   #1, S^BOOTBLOCK_IDX, J                          ; 0805
                                                18  11 0018E        BRB     6$
                                          0000GCF42  D5 00190 5$:    TSTL    ALLOC_TABLE_CNT[J]                              ; 0807
                                                11  13 00195        BEQL    6$
                                                7E  D4 00197        CLRL    -(SP)                                           ; 0808
                                          0000GCF42  DF 00199        PUSHAL  ALLOC_TABLE_LBN[J]
                                          0000GCF42  DF 0019E        PUSHAL  ALLOC_TABLE_CNT[J]
                                0000V  CF           03  FB 001A3        CALLS   #3, MAKE_POINTER
                          E0         52 00000000G  8F  F3 001A8 6$:    ACBLEQ  #IDXFILE_IDX-1, J, 5$                           ; 0805
                             0C   AE 00000000*  EF  D0 001B0        MOVL    <ALLOC_TABLE_CNT+<IDXFILE_IDX*4>>, -            ; 0816
                                                                                IDXFILE_EXT_CNT
                             08   AE 00000000*  EF  D0 001B8        MOVL    <ALLOC_TABLE_LBN+<IDXFILE_IDX*4>>, -            ; 0817
                                                                                IDXFILE_EXT_LBN
                                58        01    A7  9E 001C0        MOVAB   1(R7), R8                                       ; 0837
                                                01  DD 001C4        PUSHL   #1                                              ; 0819
                                          0C    AE  9F 001C6        PUSHAB  IDXFILE_EXT_LBN
                                          14    AE  9F 001C9        PUSHAB  IDXFILE_EXT_CNT
                                0000V  CF           03  FB 001CC        CALLS   #3, MAKE_POINTER
                                03         50  E9 001D1        BLBC    R0, 7$
                                     00D5  31  001D4        BRW     10$
                                          10    AE  9F 001D7 7$:    PUSHAB  BUFFER1                                         ; 0825
                                     0000G  CF  DD 001DA        PUSHL   HOMEBLOCK1_LBN
                                0000G  CF           02  FB 001DE        CALLS   #2, READ_BLOCK
                                  1C   AE   0102    8F  B0 001E3        MOVW    #258, BUFFER1+12                        ; 0826
                                                3A  DD 001E9        PUSHL   #58                                             ; 0827
                                          14    AE  9F 001EB        PUSHAB  BUFFER1
                                0000G  CF           02  FB 001EE        CALLS   #2, CHECKSUM2
                                7E         01FE 8F  3C 001F3        MOVZWL  #510, -(SP)                                     ; 0828
                                          14    AE  9F 001F8        PUSHAB  BUFFER1
                                0000G  CF           02  FB 001FB        CALLS   #2, CHECKSUM2
                                          10    AE  9F 00200        PUSHAB  BUFFER1                                         ; 0829
                                     0000G  CF  DD 00203        PUSHL   HOMEBLOCK1_LBN
                                6B         02  FB 00207        CALLS   #2, WRITE_BLOCK
                                56         06  D0 0020A        MOVL    #6, EXTENSION_FID                               ; 0831
                                     5C    AA  B4 0020D        CLRW    MAP_AREA                                        ; 0835
                             5E   AA         56  B0 00210        MOVW    EXTENSION_FID, MAP_AREA+2                       ; 0833
                             60   AA         56  B0 00214        MOVW    EXTENSION_FID, MAP_AREA+4                       ; 0834
                                7E   01FE 8F  3C 00218        MOVZWL  #510, -(SP)                                     ; 0836
```

```
                                  5A  DD 0021D        PUSHL   R10                                        ;
                 0000G  CF        02  FB 0021F        CALLS   #2, CHECKSUM2                              ;
                         0500     8F  BB 00224        PUSHR   #^M<R8,R10>                                ; 0837
                        6B        02  FB 00228        CALLS   #2, WRITE_BLOCK                            ;
                    02  AA        56  B0 0022B 8$:    MOVW    EXTENSION_FID, BUFFER+2                    ; 0841
                    04  AA        56  B0 0022F        MOVW    EXTENSION_FID, BUFFER+4                    ; 0842
                            64    AA  94 00233        CLRB    MAP_AREA+8                                 ; 0843
 019A  8F      00       6E        00  2C 00236        MOVC5   #0, (SP), #0, #410, BUFFER+102             ; 0844
                            66    AA     0023D                                                           ;
                                  01  DD 0023F        PUSHL   #1                                         ; 0845
                            0C    AE  9F 00241        PUSHAB  IDXFILE_EXT_LBN                            ;
                            14    AE  9F 00244        PUSHAB  IDXFILE_EXT_CNT                            ;
                 0000V  CF        03  FB 00247        CALLS   #3, MAKE_POINTER                           ;
                        59        50  D0 0024C        MOVL    R0, MAP_FULL_STATUS                        ;
                        50  01    A6  9E 0024F        MOVAB   1(R6), R0                                  ; 0846
                    5E  AA        50  B0 00253        MOVW    R0, MAP_AREA+2                             ;
                    60  AA        50  B0 00257        MOVW    R0, MAP_AREA+4                             ; 0847
                        03        59  E9 0025B        BLBC    MAP_FULL_STATUS, 9$                        ; 0848
                            5E    AA  D4 0025E        CLRL    MAP_AREA+2                                 ; 0851
                            5C    AA  96 00261 9$:    INCB    MAP_AREA                                   ; 0854
                        7E  01FE  8F  3C 00264        MOVZWL  #510, -(SP)                                ; 0855
                                  5A  DD 00269        PUSHL   R10                                        ;
                 0000G  CF        02  FB 0026B        CALLS   #2, CHECKSUM2                              ;
                                  5A  DD 00270        PUSHL   R10                                        ; 0856
                                6647  9F 00272        PUSHAB  (EXTENSION_FID)[LBN]                       ;
                        6B        02  FB 00275        CALLS   #2, WRITE_BLOCK                            ;
                                  56  D6 00278        INCL    EXTENSION_FID                              ; 0857
                                  59  D5 0027A        TSTL    MAP_FULL_STATUS                            ; 0859
                                  AD  13 0027C        BEQL    8$                                         ;
                                  5A  DD 0027E        PUSHL   R10                                        ; 0864
                         0000G    CF  DD 00280        PUSHL   IDXFILE_LBN                                ;
                 0000G  CF        02  FB 00284        CALLS   #2, READ_BLOCK                             ;
                        56        06  C2 00289        SUBL2   #6, R6                                     ; 0865
       6A      56       05 FFFFFFFF 8F F0 0028C       INSV    #-1, #5, R6, BUFFER                        ;
                                  5A  DD 00295        PUSHL   R10                                        ; 0866
                         0000G    CF  DD 00297        PUSHL   IDXFILE_LBN                                ;
                        6B        02  FB 0029B        CALLS   #2, WRITE_BLOCK                            ;
                         0500     8F  BB 0029E        PUSHR   #^M<R8,R10>                                ; 0870
                 0000G  CF        02  FB 002A2        CALLS   #2, READ_BLOCK                             ;
                            5E    AA  D4 002A7        CLRL    MAP_AREA+2                                 ; 0871
                                  13  11 002AA        BRB     11$                                        ; 0819
                        7E  01FE  8F  3C 002AC 10$:   MOVZWL  #510, -(SP)                                ; 0876
                                  5A  DD 002B1        PUSHL   R10                                        ;
                 0000G  CF        02  FB 002B3        CALLS   #2, CHECKSUM2                              ;
                         0500     8F  BB 002B8        PUSHR   #^M<R8,R10>                                ; 0877
                        6B        02  FB 002BC        CALLS   #2, WRITE_BLOCK                            ;
 019A  8F      00       6E        00  2C 002BF 11$:   MOVC5   #0, (SP), #0, #410, BUFFER+102             ; 0883
                            66    AA     002C6                                                           ;
                    02  AA 00030003 8F D0 002C8       MOVL    #196611, BUFFER+2                          ; 0884
                            64    AA  94 002D0        CLRB    MAP_AREA+8                                 ; 0887
                                  51  D4 002D3        CLRL    MAP_COUNT                                  ; 0888
                        53  0000G CF  D0 002D5        MOVL    BADBLOCK_TOTAL, R3                         ; 0889
                        50        01  CE 002DA        MNEGL   #1, J                                      ;
                                  06  11 002DD        BRB     13$                                        ;
                        51  0000GCF40 C0 002DF 12$:   ADDL2   BADBLOCK_CNT[J], MAP_COUNT                 ; 0890
               F6       50        53  F2 002E5 13$:   AOBLSS  R3, J, 12$                                 ;
                    2E  AA 0E6B0CAC 8F D0 002E9       MOVL    #241896620, IDENT_AREA                     ; 0892
```

```
                        52           01  CE 002F1          MNEGL   #1, J                                    ; 0893
                                     25  11 002F4          BRB     16$                                      ;
                  CA    8F       64  AA  91 002F6 14$:     CMPB    MAP_AREA+8, #202                         ; 0895
                                     0D  1B 002FB          BLEQU   15$                                      ;
                         007580BC    8F  DD 002FD          PUSHL   #7700668                                 ; 0896
             00000000G  00           01  FB 00303          CALLS   #1, LIB$STOP                             ;
                                     7E  D4 0030A 15$:     CLRL    -(SP)                                    ; 0897
                           0000GCF42     DF 0030C          PUSHAL  BADBLOCK_LBN[J]                          ;
                           0000GCF42     DF 00311          PUSHAL  BADBLOCK_CNT[J]                          ;
                  0000V  CF          03  FB 00316          CALLS   #3, MAKE_POINTER                         ;
              D7         52          53  F2 0031B 16$:     AOBLSS  R3, J, 14$                               ; 0893
                         7E     01FE 8F  3C 0031F          MOVZWL  #510, -(SP)                              ; 0899
                                     5A  DD 00324          PUSHL   R10                                      ;
                  0000G  CF          02  FB 00326          CALLS   #2, CHECKSUM2                            ;
                                     5A  DD 0032B          PUSHL   R10                                      ; 0900
                                 03  A7  9F 0032D          PUSHAB  3(LBN)                                   ;
                         6B          02  FB 00330          CALLS   #2, WRITE_BLOCK                          ;
  019A   8F          00          6E  00  2C 00333          MOVC5   #0, (SP), #0, #410, BUFFER+102           ; 0905
                                 66  AA     0033A                                                       ;
                  02     AA 00020002 8F  D0 0033C          MOVL    #131074, BUFFER+2                        ; 0906
                                 64  AA  94 00344          CLRB    MAP_AREA+8                               ; 0908
                  2E     AA 51780DFC 8F  D0 00347          MOVL    #1366822396, IDENT_AREA                  ; 0910
                                     7E  D4 0034F          CLRL    -(SP)                                    ; 0911
                             0000G   CF  9F 00351          PUSHAB  BITMAP_LBN                               ;
                  0C     AE          01  D0 00355          MOVL    #1, 12(SP)                               ;
                                 0C  AE  9F 00359          PUSHAB  12(SP)                                   ;
                  0000V  CF          03  FB 0035C          CALLS   #3, MAKE_POINTER                         ;
                                     7E  D4 00361          CLRL    -(SP)                                    ; 0912
          08   AE  0000G  CF         01  C1 00363          ADDL3   #1, BITMAP_LBN, 8(SP)                    ;
                                 08  AE  9F 0036A          PUSHAB  8(SP)                                    ;
          08   AE  0000G  CF         01  C3 0036D          SUBL3   #1, BITMAP_CNT, 8(SP)                    ;
                                 08  AE  9F 00374          PUSHAB  8(SP)                                    ;
                  0000V  CF          03  FB 00377          CALLS   #3, MAKE_POINTER                         ;
                         7E     01FE 8F  3C 0037C          MOVZWL  #510, -(SP)                              ; 0913
                                     5A  DD 00381          PUSHL   R10                                      ;
                  0000G  CF          02  FB 00383          CALLS   #2, CHECKSUM2                            ;
                                     5A  DD 00388          PUSHL   R10                                      ; 0914
                                 02  A7  9F 0038A          PUSHAB  2(LBN)                                   ;
                         6B          02  FB 0038D          CALLS   #2, WRITE_BLOCK                          ;
  019A   8F          00          6E  00  2C 00390          MOVC5   #0, (SP), #0, #410, BUFFER+102           ; 0919
                                 66  AA     00397                                                       ;
                                 64  AA  94 00399          CLRB    MAP_AREA+8                               ; 0920
                  02     AA 00040004 8F  D0 0039C          MOVL    #262148, BUFFER+2                        ; 0921
                  0C     AA       80 8F  88 003A4          BISB2   #128, BUFFER+12                          ; 0923
                  0E     AA          01  90 003A9          MOVB    #1, BUFFER+14                            ; 0924
                  10     AA          10  B0 003AD          MOVW    #16, BUFFER+16                           ; 0925
                  16     AA 00020000 8F  D0 003B1          MOVL    #131072, BUFFER+22                       ; 0926
          12   AA  0000G  CF         10  9C 003B9          ROTL    #16, MFD_CNT, BUFFER+18                  ; 0927
                  2E     AA C04EC04E 8F  D0 003C0          MOVL    #-1068580786, IDENT_AREA                 ; 0929
                  32     AA 1A7A0000 8F  D0 003C8          MOVL    #444203008, IDENT_AREA+4                 ; 0930
                                     7E  D4 003D0          CLRL    -(SP)                                    ; 0931
                             0000G   CF  9F 003D2          PUSHAB  MFD_LBN                                  ;
                             0000G   CF  9F 003D6          PUSHAB  MFD_CNT                                  ;
                  0000V  CF          03  FB 003DA          CALLS   #3, MAKE_POINTER                         ;
                         7E     01FE 8F  3C 003DF          MOVZWL  #510, -(SP)                              ; 0932
                                     5A  DD 003E4          PUSHL   R10                                      ;
                  0000G  CF          02  FB 003E6          CALLS   #2, CHECKSUM2                            ;
```

```
                                 5A  DD 003EB          PUSHL   R10                                     ; 0933
                              04 A7  9F 003ED          PUSHAB  4(LBN)                                  ;
                           6B    02  FB 003F0          CALLS   #2, WRITE_BLOCK                         ;
                                     04 003F3          RET                                             ; 0935

; Routine Size:  1012 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  518          0936  1 ROUTINE MAKE_POINTER (COUNT, LBN, SAVE) =
;  519          0937  1
;  520          0938  1 !++
;  521          0939  1 !
;  522          0940  1 ! FUNCTIONAL DESCRIPTION:
;  523          0941  1 !
;  524          0942  1 !       This routine appends retrieval pointers to the map area of the current
;  525          0943  1 !       file header describing the given count and LBN.
;  526          0944  1 !
;  527          0945  1 !
;  528          0946  1 ! CALLING SEQUENCE:
;  529          0947  1 !       MAKE_POINTER (ARG1, ARG2, ARG3)
;  530          0948  1 !
;  531          0949  1 ! INPUT PARAMETERS:
;  532          0950  1 !       ARG1: address of block count
;  533          0951  1 !       ARG2: address of start LBN
;  534          0952  1 !       ARG3: 0 - don't write back resulting count and LBN
;  535          0953  1 !             1 - write back resulting count and LBN
;  536          0954  1 !
;  537          0955  1 ! IMPLICIT INPUTS:
;  538          0956  1 !       BUFFER contains file header
;  539          0957  1 !
;  540          0958  1 ! OUTPUT PARAMETERS:
;  541          0959  1 !       NONE
;  542          0960  1 !
;  543          0961  1 ! IMPLICIT OUTPUTS:
;  544          0962  1 !       retrieval pointer added to header
;  545          0963  1 !
;  546          0964  1 ! ROUTINE VALUE:
;  547          0965  1 !       1 - if all went well
;  548          0966  1 !       0 - if the map area was filled
;  549          0967  1 !
;  550          0968  1 ! SIDE EFFECTS:
;  551          0969  1 !       NONE
;  552          0970  1 !
;  553          0971  1 !--
;  554          0972  1
;  555          0973  2 BEGIN
;  556          0974  2
;  557          0975  2 LOCAL
;  558          0976  2         CURRENT_COUNT,                  ! running block count
;  559          0977  2         CURRENT_LBN,                    ! running LBN
;  560          0978  2         MAP_AREA        : REF BBLOCK,   ! pointer to map area
;  561          0979  2         MAP_POINTER     : REF BBLOCK;   ! pointer to map area
;  562          0980  2
;  563          0981  2 EXTERNAL
;  564          0982  2         BUFFER          : BBLOCK;       ! buffer containing file header
;  565          0983  2
;  566          0984  2
;  567          0985  2 ! Compute the address in the file header where the pointer should go.
;  568          0986  2 ! Then determine the format of the pointer and build it.
;  569          0987  2 !
;  570          0988  2
;  571          0989  2 MAP_AREA = BUFFER + 2 * (.BUFFER[FM1$B_MPOFFSET]);
;  572          0990  2 MAP_POINTER = .MAP_AREA + FM1$C_POINTERS + .MAP_AREA[FM1$B_INUSE]*2;
;  573          0991  2 CURRENT_COUNT = ..COUNT;
;  574          0992  2 CURRENT_LBN = ..LBN;
```

```
;  575          0993  2
;  576          0994  2 DO
;  577          0995  3     BEGIN
;  578          0996  3     MAP_AREA[FM1$B_INUSE] = .MAP_AREA[FM1$B_INUSE] + 2;
;  579          0997  3     MAP_POINTER[FM1$B_HIGHLBN] = .CURRENT_LBN<16,8>;
;  580          0998  3     MAP_POINTER[FM1$B_COUNT] = MIN (.CURRENT_COUNT, 256) - 1;
;  581          0999  3     MAP_POINTER[FM1$W_LOWLBN] = .CURRENT_LBN<0,16>;
;  582          1000  3     MAP_POINTER = .MAP_POINTER + 4;
;  583          1001  3
;  584          1002  3     CURRENT_LBN = .CURRENT_LBN + MIN (.CURRENT_COUNT, 256);
;  585          1003  3     CURRENT_COUNT = .CURRENT_COUNT - MIN (.CURRENT_COUNT, 256);
;  586          1004  3
;  587          1005  3     IF .MAP_POINTER GEQA BUFFER[FH1$W_CHECKSUM]
;  588          1006  3     THEN
;  589          1007  4         BEGIN
;  590          1008  4         IF .SAVE EQL 0 THEN RETURN 0;
;  591          1009  4         .COUNT = .CURRENT_COUNT;
;  592          1010  4         .LBN = .CURRENT_LBN;
;  593          1011  4         RETURN 0;
;  594          1012  3         END;
;  595          1013  3
;  596          1014  3     END
;  597          1015  2 UNTIL .CURRENT_COUNT EQL 0;
;  598          1016  2 RETURN 1;
;  599          1017  2
;  600          1018  1 END;                                    ! end of routine MAKE_POINTER


                                001C 00000 MAKE_POINTER:
                                           .WORD   Save R2,R3,R4                                    ; 0936
                        50      0000G   CF 9A 00002            MOVZBL  BUFFER+1, R0                ; 0989
                        52    0000GCF40    3E 00007            MOVAW   BUFFER[R0], MAP_AREA        ;
                        50        08   A2 9A 0000D            MOVZBL  8(MAP_AREA), R0             ; 0990
                        51      0A A240    3E 00011            MOVAW   10(MAP_AREA)[R0], MAP_POINTER ;
                        50        04   BC D0 00016            MOVL    @COUNT, CURRENT_COUNT       ; 0991
                        54        08   BC D0 0001A            MOVL    @LBN, CURRENT_LBN           ; 0992
                    08  A2              02 80 0001E 1$:       ADDB2   #2, 8(MAP_AREA)             ; 0996
53                  54                08 EF 00022            EXTZV   #16, #8, CURRENT_LBN, R3    ; 0997
                                      61 53 90 00027          MOVB    R3, (MAP_POINTER)           ;
                                      53 50 D0 0002A          MOVL    CURRENT_COUNT, R3           ; 0998
                00000100          8F  53 D1 0002D           CMPL    R3, #256                    ;
                                      05 15 00034           BLEQ    2$                          ;
                                   53 0100   8F 3C 00036      MOVZWL  #256, R3                    ;
                01    A1              53 01 83 0003B 2$:    SUBB3   #1, R3, 1(MAP_POINTER)      ;
                            02    A1    54 B0 00040          MOVW    CURRENT_LBN, 2(MAP_POINTER) ; 0999
                                      51 04 C0 00044          ADDL2   #4, MAP_POINTER             ; 1000
                                      54 53 C0 00047          ADDL2   R3, CURRENT_LBN             ; 1002
                                      50 53 C2 0004A          SUBL2   R3, CURRENT_COUNT           ; 1003
                                   53 0000G   CF 9E 0004D     MOVAB   BUFFER+510, R3              ; 1005
                                      53 51 D1 00052          CMPL    MAP_POINTER, R3             ;
                                         0F 1F 00055          BLSSU   3$                          ;
                                      0C   AC D5 00057        TSTL    SAVE                        ; 1008
                                         12 13 0005A          BEQL    4$                          ;
                            04    BC    50 D0 0005C          MOVL    CURRENT_COUNT, @COUNT       ; 1009
```

```
                          08    BC              54  D0 00060          MOVL    CURRENT_LBN, @LBN                       ; 1010
                                                08  11 00064          BRB     4$                                      ; 1011
                                                50  D5 00066  3$:     TSTL    CURRENT_COUNT                           ; 1015
                                                B4  12 00068          BNEQ    1$                                      ;
                                50              01  D0 0006A          MOVL    #1, R0                                  ; 1016
                                                    04 0006D          RET                                             ;
                                                50  D4 0006E  4$:     CLRL    R0                                      ; 1018
                                                    04 00070          RET                                             ;

; Routine Size:  113 bytes,    Routine Base:  $CODE$ + 03F4

;  601          1019  1
;  602          1020  1 END
;  603          1021  0 ELUDOM


                                                                      .EXTRN  LIB$STOP

;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $PLIT$                        194  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       1125  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        61         0        1000        00:01.8


;                                     COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:ININD1/OBJ=OBJ$:ININD1 MSRC$:ININD1/UPDATE=(ENH$:ININD1)

; Size:         1125 code + 194 data bytes
; Run Time:       00:26.6
; Elapsed Time:   00:48.0
; Lines/CPU Min:   2303
; Lexemes/CPU-Min: 28050
; Memory Used:  293 pages
; Compilation Complete
```