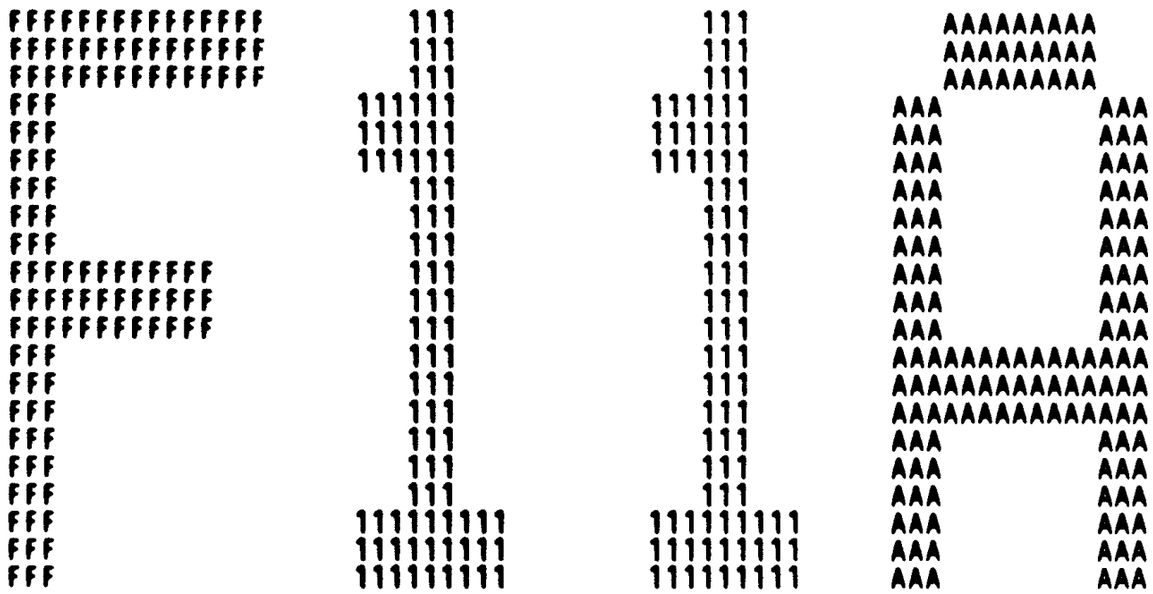
FFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
AAAAAAAAA
AAAAAAAAA
AAAAAAAAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA

**FILE**ID**DISPAT

```
DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP      AAAAAA    TTTTTTTTTT
DDDDDDDD      IIIIII      SSSSSSSS  PPPPPPPP      AAAAAA    TTTTTTTTTT
DD      DD      II      SS          PP      PP  AA      AA      TT
DD      DD      II      SS          PP      PP  AA      AA      TT
DD      DD      II      SS          PP      PP  AA      AA      TT
DD      DD      II      SS          PP      PP  AA      AA      TT
DD      DD      II        SSSSSS    PPPPPPPP    AA      AA      TT
DD      DD      II        SSSSSS    PPPPPPPP    AA      AA      TT
DD      DD      II              SS  PP          AAAAAAAAAA      TT
DD      DD      II              SS  PP          AAAAAAAAAA      TT
DD      DD      II              SS  PP          AA      AA      TT          ....
DD      DD      II              SS  PP          AA      AA      TT          ....
DDDDDDDD      IIIIII    SSSSSSSS    PP          AA      AA      TT          ....
DDDDDDDD      IIIIII    SSSSSSSS    PP          AA      AA      TT          ....

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

ENT
V04

```
;     1        0001  0 MODULE DISPAT (
;     2        0002  0                 LANGUAGE (BLISS32),
;     3        0003  0                 MAIN  = STARTUP,
;     4        0004  0                 IDENT = 'V04-000'
;     5        0005  0                 ) =
;     6        0006  1 BEGIN
;     7        0007  1
;     8        0008  1 !
;     9        0009  1 !**********************************************************************
;    10        0010  1 !*                                                                    *
;    11        0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    12        0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    13        0013  1 !*  ALL RIGHTS RESERVED.                                             *
;    14        0014  1 !*                                                                    *
;    15        0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    16        0016  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  *
;    17        0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  *
;    18        0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    19        0019  1 !*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  *
;    20        0020  1 !*  TRANSFERRED.                                                      *
;    21        0021  1 !*                                                                    *
;    22        0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  *
;    23        0023  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  *
;    24        0024  1 !*  CORPORATION.                                                      *
;    25        0025  1 !*                                                                    *
;    26        0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  *
;    27        0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.           *
;    28        0028  1 !*                                                                    *
;    29        0029  1 !*                                                                    *
;    30        0030  1 !**********************************************************************
;    31        0031  1
;    32        0032  1 !++
;    33        0033  1 !
;    34        0034  1 ! FACILITY:  F11ACP Structure Level 1
;    35        0035  1 !
;    36        0036  1 ! ABSTRACT:
;    37        0037  1 !
;    38        0038  1 !       This module is the main routine of FCP. It dequeues a request,
;    39        0039  1 !       executes it, and signals completion to the user.
;    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:  20-Dec-1976  14:33
;    50        0050  1 !
;    51        0051  1 ! MODIFIED BY:
;    52        0052  1 !
;    53        0053  1 !       V02-001 LMP0005         L. Mark Pilant,         13-Jan-1982  13:10
;    54        0054  1 !               Adding support for Cathedral windows.
;    55        0055  1 !
;    56        0056  1 !       A0100   ACG0001         Andrew C. Goldstein,  10-Oct-1978  20:01
;    57        0057  1 !               Previous revision history moved to F11A.REV
```

```
;    58          0058  1 !
;    59          0059  1 !**
;    60          0060  1
;    61          0061  1
;    62          0062  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    63          0063  1 REQUIRE 'SRC$:FCPDEF.B32';
;    64          0378  1
;    65          0379  1 !
;    66          0380  1 ! Establish the max and min function codes for the function dispatch.
;    67          0381  1 !
;    68          0382  1
;    69          0383  1 LITERAL
;    70          0384  1         LOW_FUNCTION    = MINU (
;    71          0385  1                 IO$_ACCESS,
;    72          0386  1                 IO$_CREATE,
;    73          0387  1                 IO$_DEACCESS,
;    74          0388  1                 IO$_DELETE,
;    75          0389  1                 IO$_MODIFY,
;    76          0390  1                 IO$_ACPCONTROL,
;    77          0391  1                 IO$_MOUNT
;    78          0392  1                 ),
;    79          0393  1
;    80          0394  1         HIGH_FUNCTION   = MAXU (
;    81          0395  1                 IO$_ACCESS,
;    82          0396  1                 IO$_CREATE,
;    83          0397  1                 IO$_DEACCESS,
;    84          0398  1                 IO$_DELETE,
;    85          0399  1                 IO$_MODIFY,
;    86          0400  1                 IO$_ACPCONTROL,
;    87          0401  1                 IO$_MOUNT
;    88          0402  1                 );
;    89          0403  1
;    90          0404  1 FORWARD ROUTINE
;    91          0405  1         STARTUP         : NOVALUE,
;    92          0406  1         DISPATCHER      : NOVALUE,
;    93          0407  1         MAIN_HANDLER    : NOVALUE;
```

```
;    95          0408  1 GLOBAL ROUTINE STARTUP : NOVALUE =
;    96          0409  1
;    97          0410  1 !++
;    98          0411  1 !
;    99          0412  1 ! FUNCTIONAL DESCRIPTION:
;   100          0413  1 !
;   101          0414  1 !       This routine is the startup point for FCP. It locks all of FCP
;   102          0415  1 !       into memory and then calls the dispatcher loop in exec mode.
;   103          0416  1 !
;   104          0417  1 ! CALLING SEQUENCE:
;   105          0418  1 !       UNDEFINED
;   106          0419  1 !
;   107          0420  1 ! INPUT PARAMETERS:
;   108          0421  1 !       NONE
;   109          0422  1 !
;   110          0423  1 ! IMPLICIT INPUTS:
;   111          0424  1 !       NONE
;   112          0425  1 !
;   113          0426  1 ! OUTPUT PARAMETERS:
;   114          0427  1 !       NONE
;   115          0428  1 !
;   116          0429  1 ! IMPLICIT OUTPUTS:
;   117          0430  1 !       NONE
;   118          0431  1 !
;   119          0432  1 ! ROUTINE VALUE:
;   120          0433  1 !       NONE
;   121          0434  1 !
;   122          0435  1 ! SIDE EFFECTS:
;   123          0436  1 !       FCP locked into memory, dispatcher started.
;   124          0437  1 !
;   125          0438  1 !--
;   126          0439  1
;   127          0440  2 BEGIN
;   128          0441  2
;   129          0442  2
;   130          0443  2 EXEC_CALL (DISPATCHER);
;   131          0444  2
;   132          0445  1 END;                                    ! end of routine STARTUP

                                                        .TITLE  DISPAT
                                                        .IDENT  \V04-000\

                                                        .EXTRN  SYS$CMEXEC

                                                        .PSECT  $CODE$,NOWRT,2

                                          0000 00000    .ENTRY  STARTUP, Save nothing                   ; 0408
                                       7E   D4 00002    CLRL    -(SP)                                   ; 0443
                                       5E   DD 00004    PUSHL   SP
                                  0000V CF   9F 00006    PUSHAB  DISPATCHER
                       00000000G   9F      03   FB 0000A    CALLS   #3, @#SYS$CMEXEC
                                           04 00011    RET                                             ; 0445

; Routine Size:  18 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  134     0446  1 GLOBAL ROUTINE DISPATCHER : NOVALUE =
;  135     0447  1
;  136     0448  1 !++
;  137     0449  1 !
;  138     0450  1 ! FUNCTIONAL DESCRIPTION:
;  139     0451  1 !
;  140     0452  1 !       This routine is the main routine of FCP. It dequeues a request,
;  141     0453  1 !       executes it, and signals completion to the user.
;  142     0454  1 !
;  143     0455  1 ! CALLING SEQUENCE:
;  144     0456  1 !       DISPATCHER ()
;  145     0457  1 !
;  146     0458  1 ! INPUT PARAMETERS:
;  147     0459  1 !       NONE
;  148     0460  1 !
;  149     0461  1 ! IMPLICIT INPUTS:
;  150     0462  1 !       NONE
;  151     0463  1 !
;  152     0464  1 ! OUTPUT PARAMETERS:
;  153     0465  1 !       NONE
;  154     0466  1 !
;  155     0467  1 ! IMPLICIT OUTPUTS:
;  156     0468  1 !       NONE
;  157     0469  1 !
;  158     0470  1 ! ROUTINE VALUE:
;  159     0471  1 !       NONE
;  160     0472  1 !
;  161     0473  1 ! SIDE EFFECTS:
;  162     0474  1 !       FCP functions executed
;  163     0475  1 !
;  164     0476  1 !--
;  165     0477  1
;  166     0478  2 BEGIN
;  167     0479  2
;  168     0480  2 LOCAL
;  169     0481  2         FUNCTION;                       ! function being executed
;  170     0482  2
;  171     0483  2 EXTERNAL
;  172     0484  2         IO_PACKET       : REF BBLOCK;   ! current I/O packet
;  173     0485  2
;  174     0486  2 EXTERNAL ROUTINE
;  175     0487  2         INIT_FCP,                       ! one time initialization
;  176     0488  2         INIT_COMMON,                    ! per call initialization
;  177     0489  2         PMS_START,                      ! start performance metering
;  178     0490  2         PMS_END,                        ! end performance metering
;  179     0491  2         GET_REQUEST,                    ! get next I/O request
;  180     0492  2         READ_WRITEVB,                   ! process read/write virtual
;  181     0493  2         ACCESS,                         ! ACCESS function routine
;  182     0494  2         CREATE,                         ! CREATE function routine
;  183     0495  2         DEACCESS,                       ! DEACCESS function routine
;  184     0496  2         DELETE,                         ! DELETE function routine
;  185     0497  2         MODIFY,                         ! MODIFY function routine
;  186     0498  2         ACPCONTROL,                     ! ACPCONTROL function routine
;  187     0499  2         MOUNT,                          ! MOUNT function routine
;  188     0500  2         ERR_CLEANUP,                    ! error cleanup routine
;  189     0501  2         CLEANUP,                        ! general cleanup routine
;  190     0502  2         IO_DONE;                        ! I/O completion processing
```

```
;  191          0503  2
;  192          0504  2
;  193          0505  2 ! Do the one time initialization. Then for each request, do the per
;  194          0506  2 ! request initialization, get the next request, and process it. If
;  195          0507  2 ! the request fails, call the error cleanup before returning
;  196          0508  2 ! completion.
;  197          0509  2 !
;  198          0510  2
;  199          0511  2 !       ENABLE MAIN_HANDLER;
;  200          0512  3 BEGIN
;  201          0513  3 BUILTIN FP;
;  202          0514  3 .FP = MAIN_HANDLER;
;  203          0515  2 END;
;  204          0516  2 KERNEL_CALL (INIT_FCP);
;  205          0517  2
;  206          0518  2 WHILE 1 DO
;  207          0519  3     BEGIN
;  208          0520  3     PMS_START ();
;  209          0521  3     INIT_COMMON ();
;  210          0522  3     IO_PACKET = KERNEL_CALL (GET_REQUEST);
;  211          0523  3     FUNCTION = .IO_PACKET[IRP$V_FCODE];
;  212          0524  3
;  213          0525  3     IF .FUNCTION EQL IO$_READPBLK
;  214          0526  3     OR .FUNCTION EQL IO$_WRITEPBLK
;  215          0527  3     THEN
;  216          0528  4         BEGIN
;  217          0529  4         IF NOT READ_WRITEVB ()
;  218          0530  4         THEN
;  219          0531  5             BEGIN
;  220          0532  5             IF ERR_CLEANUP () THEN CLEANUP () ELSE (ERR_CLEANUP (); CLEANUP ());
;  221          0533  5             PMS_END ();
;  222          0534  5             KERNEL_CALL (IO_DONE, .IO_PACKET);
;  223          0535  4             END;
;  224          0536  4         END
;  225          0537  3     ELSE
;  226          0538  4         BEGIN
;  227          0539  4
;  228          0540  4         IF
;  229          0541  5             (
;  230          0542  5             CASE .FUNCTION FROM LOW_FUNCTION TO HIGH_FUNCTION OF
;  231          0543  5                 SET
;  232          0544  5                 [IO$_ACCESS]:   ACCESS ();
;  233          0545  5                 [IO$_CREATE]:   CREATE ();
;  234          0546  5                 [IO$_DEACCESS]: DEACCESS ();
;  235          0547  5                 [IO$_DELETE]:   DELETE ();
;  236          0548  5                 [IO$_MODIFY]:   MODIFY ();
;  237          0549  5                 [IO$_ACPCONTROL]: ACPCONTROL ();
;  238          0550  5                 [IO$_MOUNT]:    MOUNT ();
;  239          0551  5                 [INRANGE]:      (ERR_STATUS (SS$_ILLIOFUNC); 0);
;  240          0552  5                 [OUTRANGE]:     (ERR_STATUS (SS$_ILLIOFUNC); 0);
;  241          0553  5                 TES
;  242          0554  5             )
;  243          0555  4         THEN                            ! successful completion
;  244          0556  5             BEGIN
;  245          0557  5             IF NOT CLEANUP () THEN (ERR_CLEANUP (); CLEANUP ());
;  246          0558  5             END
;  247          0559  4         ELSE                            ! error completion
```

```
;  248          0560  5              BEGIN
;  249          0561  5              IF ERR_CLEANUP () THEN CLEANUP () ELSE (ERR_CLEANUP (); CLEANUP ());
;  250          0562  4              END;
;  251          0563  4
;  252          0564  4          PMS_END ();
;  253          0565  4          KERNEL_CALL (IO_DONE, .IO_PACKET);
;  254          0566  4          END
;  255          0567  2      END;
;  256          0568  2
;  257          0569  1 END;                                    ! end of routine DISPATCHER


                                                        .EXTRN  IO_PACKET, INIT_FCP
                                                        .EXTRN  INIT_COMMON, PMS_START
                                                        .EXTRN  PMS_END, GET_REQUEST
                                                        .EXTRN  READ_WRITEVB, ACCESS
                                                        .EXTRN  CREATE, DEACCESS
                                                        .EXTRN  DELETE, MODIFY, ACPCONTROL
                                                        .EXTRN  MOUNT, ERR_CLEANUP
                                                        .EXTRN  CLEANUP, IO_DONE
                                                        .EXTRN  SYS$CMKRNL, USER_STATUS

                                  001C 00000    .ENTRY  DISPATCHER, Save R2,R3,R4                       ; 0446
                        54 00000000G  00  9E 00002  MOVAB   USER_STATUS, R4                             ;
                        53 00000000G  9F  9E 00009  MOVAB   @#SYS$CMKRNL, R3                           ;
                        6D      0000V CF  9E 00010  MOVAB   MAIN_HANDLER, (FP)                          ; 0514
                                      7E  D4 00015  CLRL    -(SP)                                     ; 0516
                                      5E  DD 00017  PUSHL   SP                                        ;
                                0000G CF  9F 00019  PUSHAB  INIT_FCP                                    ;
                                   63     03  FB 0001D  CALLS   #3, SYS$CMKRNL                         ;
                             0000G CF     00  FB 00020 1$:  CALLS   #0, PMS_START                      ; 0520
                             0000G CF     00  FB 00025      CALLS   #0, INIT_COMMON                   ; 0521
                                      7E  D4 0002A  CLRL    -(SP)                                     ; 0522
                                      5E  DD 0002C  PUSHL   SP                                        ;
                                0000G CF  9F 0002E  PUSHAB  GET_REQUEST                               ;
                                   63     03  FB 00032  CALLS   #3, SYS$CMKRNL                         ;
                             0000G CF     50  D0 00035  MOVL    R0, IO_PACKET                         ;
    52        20    A0                06     00  EF 0003A  EXTZV   #0, #6, 32(R0), FUNCTION           ; 0523
                                   0C     52  D1 00040  CMPL    FUNCTION, #12                         ; 0525
                                          05  13 00043  BEQL    2$                                    ;
                                   0B     52  D1 00045  CMPL    FUNCTION, #11                         ; 0526
                                          0A  12 00048  BNEQ    3$                                    ;
                             0000G CF     00  FB 0004A 2$:  CALLS   #0, READ_WRITEVB                  ; 0529
                                   CE     50  E8 0004F  BLBS    R0, 1$                                ;
                                          5D  11 00052  BRB     15$                                   ; 0532
                      07                32     52  CF 00054 3$:  CASEL   FUNCTION, #50, #7             ; 0542
0027                0020              0019           0012      00058 4$:  .WORD   5$-4$,-              ;
003C                0035              0046           002E      00060              6$-4$,-              ;
                                                                                  7$-4$,-              ;
                                                                                  8$-4$,-              ;
                                                                                  9$-4$,-              ;
                                                                                  13$-4$,-             ;
                                                                                  10$-4$,-             ;
                                                                                  11$-4$               ;
                                          34  11 00068      BRB     13$                                   ; 0552
                             0000G CF     00  FB 0006A 5$:  CALLS   #0, ACCESS                        ; 0544
```

```
                                    28  11 0006F          BRB     12$
                0000G   CF          00  FB 00071 6$:     CALLS   #0, CREATE                      ; 0545
                                    21  11 00076          BRB     12$
                0000G   CF          00  FB 00078 7$:     CALLS   #0, DEACCESS                    ; 0546
                                    1A  11 0007D          BRB     12$
                0000G   CF          00  FB 0007F 8$:     CALLS   #0, DELETE                      ; 0547
                                    13  11 00084          BRB     12$
                0000G   CF          00  FB 00086 9$:     CALLS   #0, MODIFY                      ; 0548
                                    0C  11 0008B          BRB     12$
                0000G   CF          00  FB 0008D 10$:    CALLS   #0, ACPCONTROL                  ; 0549
                                    05  11 00092          BRB     12$
                0000G   CF          00  FB 00094 11$:    CALLS   #0, MOUNT                       ; 0550
                        15          50  E9 00099 12$:    BLBC    R0, 15$
                                    09  11 0009C          BRB     14$
                        10          64  E9 0009E 13$:    BLBC    USER_STATUS, 15$                ; 0551
                        64      F4  8F  9B 000A1          MOVZBW  #244, USER_STATUS
                                    0A  11 000A5          BRB     15$
                0000G   CF          00  FB 000A7 14$:    CALLS   #0, CLEANUP                     ; 0557
                        14          50  E8 000AC          BLBS    R0, 18$
                                    08  11 000AF          BRB     16$
                0000G   CF          00  FB 000B1 15$:    CALLS   #0, ERR_CLEANUP                 ; 0561
                        05          50  E8 000B6          BLBS    R0, 17$
                0000G   CF          00  FB 000B9 16$:    CALLS   #0, ERR_CLEANUP
                0000G   CF          00  FB 000BE 17$:    CALLS   #0, CLEANUP
                0000G   CF          00  FB 000C3 18$:    CALLS   #0, PMS_END                     ; 0564
                            0000G   CF  DD 000C8          PUSHL   IO_PACKET                       ; 0565
                                    01  DD 000CC          PUSHL   #1
                                    5E  DD 000CE          PUSHL   SP
                            0000G   CF  9F 000D0          PUSHAB  IO_DONE
                        63              04  FB 000D4          CALLS   #4, SYS$CMKRNL
                                    FF46  31 000D7          BRW     1$                          ; 0525

; Routine Size:  218 bytes,    Routine Base:  $CODE$ + 0012

;  258          0570  1 GLOBAL ROUTINE MAIN_HANDLER (SIGNAL, MECHANISM) : NOVALUE =
;  259          0571  1
;  260          0572  1 !++
;  261          0573  1 !
;  262          0574  1 ! FUNCTIONAL DESCRIPTION:
;  263          0575  1 !
;  264          0576  1 !       This routine is the main level condition handler. It stores the
;  265          0577  1 !       condition value (FCP error code) in the user status block, unwinds
;  266          0578  1 !       and returns from the function that was executing.
;  267          0579  1 !
;  268          0580  1 ! CALLING SEQUENCE:
;  269          0581  1 !       MAIN_HANDLER (ARG1, ARG2)
;  270          0582  1 !
;  271          0583  1 ! INPUT PARAMETERS:
;  272          0584  1 !       ARG1: address of signal array
;  273          0585  1 !       ARG2: address of mechanism array
;  274          0586  1 !
;  275          0587  1 ! IMPLICIT INPUTS:
;  276          0588  1 !       NONE
;  277          0589  1 !
;  278          0590  1 ! OUTPUT PARAMETERS:
;  279          0591  1 !       NONE
```

```
;  280          0592  1 !
;  281          0593  1 ! IMPLICIT OUTPUTS:
;  282          0594  1 !       USER_STATUS: receives signal code
;  283          0595  1 !
;  284          0596  1 ! ROUTINE VALUE:
;  285          0597  1 !       NONE
;  286          0598  1 !
;  287          0599  1 ! SIDE EFFECTS:
;  288          0600  1 !       stack unwound to main level to return to dispatcher
;  289          0601  1 !
;  290          0602  1 !--
;  291          0603  1
;  292          0604  2 BEGIN
;  293          0605  2
;  294          0606  2 MAP
;  295          0607  2         SIGNAL          : REF BBLOCK,   ! signal array arg
;  296          0608  2         MECHANISM       : REF BBLOCK;   ! mechanism array arg
;  297          0609  2
;  298          0610  2 EXTERNAL
;  299          0611  2         USER_STATUS     : WORD;         ! I/O status to user
;  300          0612  2
;  301          0613  2 EXTERNAL ROUTINE
;  302          0614  2         SYS$UNWIND      : ADDRESSING_MODE (ABSOLUTE);
;  303          0615  2
;  304          0616  2
;  305          0617  2 ! Check the signal code. The only permissible ones are SS$_UNWIND, which
;  306          0618  2 ! is ignored, and SS$_CMODUSER. The error status is the 16 bit CHMU code.
;  307          0619  2 ! If the error value is non-zero, store it in the user status (zero
;  308          0620  2 ! means just exit). Set up a return value of 0, unwind to the current
;  309          0621  2 ! depth, and return, causing the invoked function to return with failure
;  310          0622  2 ! to the dispatcher.
;  311          0623  2 !
;  312          0624  2
;  313          0625  2 IF .SIGNAL[CHF$L_SIG_NAME] EQL SS$_UNWIND THEN RETURN;
;  314          0626  2 IF .SIGNAL[CHF$L_SIG_NAME] NEQ SS$_CMODUSER
;  315          0627  2 THEN BUG_CHECK (UNXSIGNAL, FATAL, 'Unexpected signal name in ACP');
;  316          0628  2
;  317          0629  2 IF .SIGNAL[CHF$L_SIG_ARG1] NEQ 0
;  318          0630  2 THEN USER_STATUS = .SIGNAL[CHF$L_SIG_ARG1];
;  319          0631  2
;  320          0632  2 MECHANISM[CHF$L_MCH_SAVR0] = 0;
;  321          0633  2
;  322          0634  2 SYS$UNWIND (MECHANISM[CHF$L_MCH_DEPTH], 0);
;  323          0635  2
;  324          0636  2 RETURN;
;  325          0637  2
;  326          0638  1 END;                                    ! end of routine MAIN_HANDLER


                                                        .EXTRN  SYS$UNWIND, BUG$_UNXSIGNAL

                                                0000 00000      .ENTRY  MAIN_HANDLER, Save nothing      ; 0570
                        50      04      AC      D0 00002        MOVL    SIGNAL, R0                      ; 0625
            00000920    8F      04      A0      D1 00006        CMPL    4(R0), #2336                    ;
                                                30  13 0000E    BEQL    3$                              ;
            00000424    8F      04      A0      D1 00010        CMPL    4(R0), #1060                    ; 0626
```

```
                                04  13 00018          BEQL    1$                                                      ;
                                  FEFF 0001A          BUGW                                                            ; 0627
                                 0000* 0001C          .WORD   <BUG$_UNXSIGNAL!4>                                      ;
                          50    04    AC  D0 0001E 1$:  MOVL    SIGNAL, R0                                              ; 0629
                                08    A0  D5 00022      TSTL    8(R0)                                                   ;
                                      06  13 00025      BEQL    2$                                                      ;
                    0000G   CF    08    A0  B0 00027      MOVW    8(R0), USER_STATUS                                      ; 0630
                          50    08    AC  D0 0002D 2$:  MOVL    MECHANISM, R0                                           ; 0632
                                0C    A0  D4 00031      CLRL    12(R0)                                                  ;
                                7E    D4 00034          CLRL    -(SP)                                                   ; 0634
                                08    A0  9F 00036      PUSHAB  8(R0)                                                   ;
            00000000G   9F            02  FB 00039      CALLS   #2, @#SYS$UNWIND                                        ;
                                          04 00040 3$:  RET                                                           ; 0638

; Routine Size:  65 bytes,    Routine Base:  $CODE$ + 00EC

;  327          0639  1
;  328          0640  1 END
;  329          0641  0 ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                         301  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         18         0      1000          00:01.9



;                                       COMMAND QUALIFIERS

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

; Size:         301 code + 0 data bytes
; Run Time:       00:09.6
; Elapsed Time:   00:26.7
; Lines/CPU Min:   4010
; Lexemes/CPU-Min: 12938
```

```
; Memory Used:  117 pages
; Compilation Complete
```

EXTFCB.LIS

DELFIL.LIS

DIRGET.LIS

EXTIDX.LIS

IODONE.LIS

LOCKDN.LIS

ENTER.LIS

GETREQ.LIS

GETTIM.LIS

DISPAT.LIS

INIFCP.LIS

DIRFCB.LIS

EXTHDR.LIS

DIRSCN.LIS

LOGDEL.LIS

LOCKDB.LIS

FIND.LIS

GETFIB.LIS

DIRACC.LIS

INIFCB.LIS

EXTDIR.LIS

EXTEND.LIS