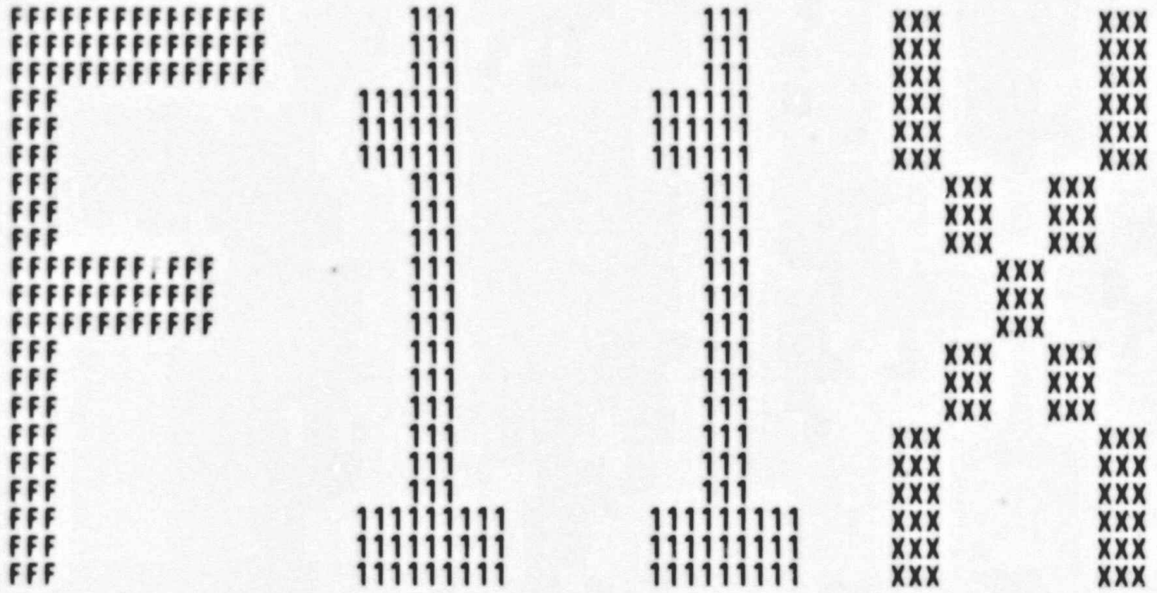

_$2

Symb

IOCS
IO_C
IO_C
IO_D
IO_F
IO_S
KICL

KILL
KILL
LB_B
LB_D
LB_F
LB_K
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK

```
  CCCCCCCC  RRRRRRR    EEEEEEEEEE  FFFFFFFFFF    CCCCCCCC  BBBBBBBB
  CCCCCCCC  RRRRRRRR   EEEEEEEEEE  FFFFFFFFFF    CCCCCCCC  BBBBBBBB
CC          RR      RR EE          FF          CC          BB      BB
CC          RR      RR EE          FF          CC          BB      BB
CC          RR      RR EE          FF          CC          BB      BB
CC          RR      RR EE          FF          CC          BB      BB
CC          RRRRRRRR   EEEEEEEE    FFFFFFFF    CC          BBBBBBBB
CC          RRRRRRRR   EEEEEEEE    FFFFFFFF    CC          BBBBBBBB
CC          RR  RR     EE          FF          CC          BB      BB
CC          RR  RR     EE          FF          CC          BB      BB
CC          RR    RR   EE          FF          CC          BB      BB      ....
CC          RR    RR   EE          FF          CC          BB      BB      ....
  CCCCCCCC  RR      RR EEEEEEEEEE  FF            CCCCCCCC  BBBBBBBB        ....
  CCCCCCCC  RR      RR EEEEEEEEEE  FF            CCCCCCCC  BBBBBBBB        ....

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 CREFCB (
;     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:  F11ACP Structure Level 2
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !
;    37        0037  1 !       These routines create and initialize a file control block
;    38        0038  1 !       from the given file header.
;    39        0039  1 !
;    40        0040  1 ! ENVIRONMENT:
;    41        0041  1 !
;    42        0042  1 !       STARLET operating system, including privileged system services
;    43        0043  1 !       and internal exec routines. These routines must be called in
;    44        0044  1 !       kernel mode.
;    45        0045  1 !
;    46        0046  1 !
;    47        0047  1 !--
;    48        0048  1 !
;    49        0049  1 !
;    50        0050  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  14-Dec-1976  16:48
;    51        0051  1 !
;    52        0052  1 ! MODIFIED BY:
;    53        0053  1 !
;    54        0054  1 !       V03-008 CDS0004         Christian D. Saether    30-Aug-1984
;    55        0055  1 !               Add optional PRIMFCB argument to CREATE_FCB routine.
;    56        0056  1 !
;    57        0057  1 !       V03-007 CDS0003         Christian D. Saether    14-Aug-1984
```

```
;    58          0058  1 !               Have REBLD_PRIM_FCB routine mark windows incomplete.
;    59          0059  1 !
;    60          0060  1 !       V03-006 CDS0002         Christian D. Saether     1-Aug-1984
;    61          0061  1 !               Add REBLD_PRIM_FCB routine.
;    62          0062  1 !
;    63          0063  1 !       V03-005 LMP0275         L. Mark Pilant,          12-Jul-1984  14:42
;    64          0064  1 !               Initialize the ACL info in the ORB to be a null descriptor
;    65          0065  1 !               list rather than an empty queue.  This avoids the overhead
;    66          0066  1 !               of locking and unlocking the ACL mutex, only to find out
;    67          0067  1 !               that the ACL was empty.
;    68          0068  1 !
;    69          0069  1 !       V03-004 LMP0221         L. Mark Pilant,          3-Apr-1984  9:16
;    70          0070  1 !               Add support for an ORB in the FCB.
;    71          0071  1 !
;    72          0072  1 !       V03-003 CDS0001         Christian D. Saether     29-Dec-1983
;    73          0073  1 !               Use L_NORM linkage and BIND_COMMON macro.
;    74          0074  1 !
;    75          0075  1 !       V03-002 LMP0059         L. Mark Pilant,          27-Dec-1982  9:06
;    76          0076  1 !               Make FCB handling consistant with file header existance.
;    77          0077  1 !
;    78          0078  1 !       V03-001 LMP0036         L. Mark Pilant,          29-Jul-1982  13:15
;    79          0079  1 !               Set up the initial ACL segment queue.
;    80          0080  1 !
;    81          0081  1 !       V02-003 ACG0241         Andrew C. Goldstein,     11-Dec-1981  23:01
;    82          0082  1 !               Use common code in INIFC2 to set up FCB
;    83          0083  1 !
;    84          0084  1 !       V02-002 ACG0167         Andrew C. Goldstein,     16-Apr-1980  19:25
;    85          0085  1 !               Previous revision history moved to F11B.REV
;    86          0086  1 !**
;    87          0087  1
;    88          0088  1
;    89          0089  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    90          0090  1 REQUIRE 'SRC$:FCPDEF.B32';
;    91          1081  1
;    92          1082  1
;    93          1083  1 FORWARD ROUTINE
;    94          1084  1         CREATE_FCB      : L_NORM,         ! create a file ontrol block
;    95          1085  1         UPDATE_FCB      : L_NORM NOVALUE; ! update contents of primary FCB
```

```
;    97          1086  1 GLOBAL ROUTINE CREATE_FCB (HEADER, PRIMFCB) : L_NORM =
;    98          1087  1
;    99          1088  1 !++
;   100          1089  1 !
;   101          1090  1 ! FUNCTIONAL DESCRIPTION:
;   102          1091  1 !
;   103          1092  1 !       This routine creates an FCB and initializes it according to
;   104          1093  1 !       the given file header.
;   105          1094  1 !
;   106          1095  1 ! CALLING SEQUENCE:
;   107          1096  1 !       CREATE_FCB (ARG1)
;   108          1097  1 !
;   109          1098  1 ! INPUT PARAMETERS:
;   110          1099  1 !       ARG1: address of file header
;   111          1100  1 !       ARG2: optional arg to specify primary fcb
;   112          1101  1 !
;   113          1102  1 ! IMPLICIT INPUTS:
;   114          1103  1 !       NONE
;   115          1104  1 !
;   116          1105  1 ! OUTPUT PARAMETERS:
;   117          1106  1 !       NONE
;   118          1107  1 !
;   119          1108  1 ! IMPLICIT OUTPUTS:
;   120          1109  1 !       NONE
;   121          1110  1 !
;   122          1111  1 ! ROUTINE VALUE:
;   123          1112  1 !       ADDRESS OF FCB
;   124          1113  1 !
;   125          1114  1 ! SIDE EFFECTS:
;   126          1115  1 !       FCB created and initialized
;   127          1116  1 !
;   128          1117  1 !--
;   129          1118  1
;   130          1119  2 BEGIN
;   131          1120  2
;   132          1121  2 MAP
;   133          1122  2         HEADER          : REF BBLOCK;   ! file header argument
;   134          1123  2
;   135          1124  2 LOCAL
;   136          1125  2         FCB             : REF BBLOCK,   ! address of FCB created
;   137          1126  2         FCB_ORB         : REF BBLOCK;   ! address of ORB created
;   138          1127  2
;   139          1128  2 BIND_COMMON;
;   140          1129  2
;   141          1130  2 EXTERNAL ROUTINE
;   142          1131  2         ALLOCATE        : L_NORM,       ! allocate dynamic memory
;   143          1132  2         INIT_FCB2       : L_NORM;       ! initialize contents of FCB
;   144          1133  2
;   145          1134  2 ! Allocate an FCB sized and typed block. Then use the common routine
;   146          1135  2 ! to init it.
;   147          1136  2 !
;   148          1137  2
;   149          1138  2 FCB = ALLOCATE (FCB$C_LENGTH, FCB_TYPE);
;   150          1139  2 FCB[FCB$L_WLFL] = FCB[FCB$L_WLFL];
;   151          1140  2 FCB[FCB$L_WLBL] = FCB[FCB$L_WLFL];
;   152          1141  2 FCB[FCB$L_STVBN] = 1;                       ! init start VBN to 1
;   153          1142  2
```

```
;  154          1143  2 ! Now for the ORB initialzation.
;  155          1144  2
;  156          1145  2 FCB_ORB = FCB[FCB$R_ORB];
;  157          1146  2 FCB_ORB[ORB$W_SIZE] = ORB$C_LENGTH;
;  158          1147  2 FCB_ORB[ORB$B_TYPE] = DYN$C_ORB;
;  159          1148  2
;  160          1149  2 ! Common initialization.
;  161          1150  2
;  162          1151  2 IF ACTUALCOUNT EQL 2
;  163          1152  2 THEN
;  164          1153  2     INIT_FCB2 (.FCB, .HEADER, .PRIMFCB)
;  165          1154  2 ELSE
;  166          1155  2     INIT_FCB2 (.FCB, .HEADER);
;  167          1156  2
;  168          1157  2 INSQUE (.FCB, .CURRENT_VCB[VCB$L_FCBBL]);       ! Link into queue
;  169          1158  2
;  170          1159  2 RETURN .FCB;
;  171          1160  2
;  172          1161  1 END;                                    ! end of routine CREATE_FCB

                                                        .TITLE  CREFCB
                                                        .IDENT  \V04-000\

                                                        .EXTRN  ALLOCATE, INIT_FCB2

                                                        .PSECT  $CODE$,NOWRT,2

                                        0004 00000      .ENTRY  CREATE_FCB, Save R2                     ; 1086
                                    7E  D4   00002      CLRL    -(SP)                                   ; 1138
                                7E  B4   8F  9A 00004      MOVZBL  #180, -(SP)                         ;
                            0000G   CF       02  FB 00008      CALLS   #2, ALLOCATE                    ;
                                    52       50  D0 0000D      MOVL    R0, FCB                         ;
                              10    A2   10  A2  9E 00010      MOVAB   16(FCB), 16(FCB)                ; 1139
                              14    A2   10  A2  9E 00015      MOVAB   16(R2), 20(FCB)                 ; 1140
                              2C    A2       01  D0 0001A      MOVL    #1, 44(FCB)                     ; 1141
                                    50   58  A2  9E 0001E      MOVAB   88(R2), FCB_ORB                 ; 1145
                              08    A0   58  8F  9B 00022      MOVZBW  #88, 8(FCB_ORB)                 ; 1146
                              0A    A0   49  8F  90 00027      MOVB    #73, 10(FCB_ORB)                ; 1147
                                    02       6C  91 0002C      CMPB    (AP), #2                        ; 1151
                                             0D  12 0002F      BNEQ    1$                              ;
                                    7E   04  AC  7D 00031      MOVQ    HEADER, -(SP)                   ; 1153
                                         52  DD  00035      PUSHL   FCB                             ;
                            0000G   CF       03  FB 00037      CALLS   #3, INIT_FCB2                   ;
                                             0A  11 0003C      BRB     2$                              ;
                                         04  AC  DD 0003E 1$:  PUSHL   HEADER                         ; 1155
                                         52  DD  00041      PUSHL   FCB                             ;
                            0000G   CF       02  FB 00043      CALLS   #2, INIT_FCB2                   ;
                                    50   98  AA  D0 00048 2$:  MOVL    -104(BASE), R0                  ; 1157
                              04    B0       62  0E 0004C      INSQUE  (FCB), @4(R0)                   ;
                                    50       52  D0 00050      MOVL    FCB, R0                         ; 1159
                                             04  00053      RET                                     ; 1161

; Routine Size:  84 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  174          1162  1 GLOBAL ROUTINE UPDATE_FCB (HEADER) : L_NORM NOVALUE =
;  175          1163  1
;  176          1164  1 !++
;  177          1165  1 !
;  178          1166  1 ! FUNCTIONAL DESCRIPTION:
;  179          1167  1 !
;  180          1168  1 !       This routine updates the file attributes of the file's primary FCB,
;  181          1169  1 !       if any, with the file attributes of the given header. The file size
;  182          1170  1 !       is preserved.
;  183          1171  1 !
;  184          1172  1 !
;  185          1173  1 ! CALLING SEQUENCE:
;  186          1174  1 !       UPDATE_FCB (ARG1)
;  187          1175  1 !
;  188          1176  1 ! INPUT PARAMETERS:
;  189          1177  1 !       ARG1: address of file header
;  190          1178  1 !
;  191          1179  1 ! IMPLICIT INPUTS:
;  192          1180  1 !       NONE
;  193          1181  1 !
;  194          1182  1 ! OUTPUT PARAMETERS:
;  195          1183  1 !       NONE
;  196          1184  1 !
;  197          1185  1 ! IMPLICIT OUTPUTS:
;  198          1186  1 !       PRIMARY_FCB: address of file FCB or 0
;  199          1187  1 !
;  200          1188  1 ! ROUTINE VALUE:
;  201          1189  1 !       NONE
;  202          1190  1 !
;  203          1191  1 ! SIDE EFFECTS:
;  204          1192  1 !       FCB is updated if it exists
;  205          1193  1 !
;  206          1194  1 !--
;  207          1195  1
;  208          1196  2 BEGIN
;  209          1197  2
;  210          1198  2 MAP
;  211          1199  2         HEADER          : REF BBLOCK;   ! file header arg
;  212          1200  2
;  213          1201  2 BIND_COMMON;
;  214          1202  2
;  215          1203  2 EXTERNAL ROUTINE
;  216          1204  2         FILL_FCB        : L_NORM;       ! fill in FCB from header
;  217          1205  2
;  218          1206  2 ! All we do is call the common routine for the primary FCB.
;  219          1207  2 !
;  220          1208  2
;  221          1209  2 IF .PRIMARY_FCB NEQ 0
;  222          1210  2 THEN
;  223          1211  3     BEGIN
;  224          1212  3     CLEANUP_FLAGS[CLF_NOBUILD] = 1;
;  225          1213  3     FILL_FCB (.PRIMARY_FCB, .HEADER);
;  226          1214  2     END;
;  227          1215  2
;  228          1216  1 END;                                    ! end of routine UPDATE_FCB
```

```
                                                        .EXTRN  FILL_FCB

                                        0000 00000      .ENTRY  UPDATE_FCB, Save nothing                        ; 1162
                                    08  AA   D5 00002   TSTL    8(BASE)                                         ; 1209
                                        0F   13 00005   BEQL    1$                                              ;
                                    01  AA   08 88 00007   BISB2   #8, 1(BASE)                                  ; 1212
                                        04  AC   DD 0000B   PUSHL   HEADER                                      ; 1213
                                        08  AA   DD 0000E   PUSHL   8(BASE)                                     ;
                                0000G   CF       02 FB 00011   CALLS   #2, FILL_FCB                             ;
                                                 04 00016 1$:  RET                                              ; 1216

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 0054

;  229          1217  1
```

```
;  231          1218  1 GLOBAL ROUTINE REBLD_PRIM_FCB (PFCB, HEADER) : L_NORM NOVALUE =
;  232          1219  1
;  233          1220  1 !++
;  234          1221  1 !
;  235          1222  1 ! Functional Description:
;  236          1223  1 !
;  237          1224  1 ! This routines removes ACL's, if any, and deletes the extension
;  238          1225  1 ! FCB chain, if any.
;  239          1226  1 ! The access lock is rearmed if the fcb is stale.
;  240          1227  1 ! It then rebuilds the fcb from the header, if present.
;  241          1228  1 !
;  242          1229  1 !--
;  243          1230  1
;  244          1231  2 BEGIN
;  245          1232  2
;  246          1233  2 MAP
;  247          1234  2         PFCB    : REF BBLOCK,
;  248          1235  2         HEADER  : REF BBLOCK;
;  249          1236  2
;  250          1237  2 BIND_COMMON;
;  251          1238  2
;  252          1239  2 EXTERNAL ROUTINE
;  253          1240  2         ACL_DELETEACL,
;  254          1241  2         CONV_ACCLOCK    : L_NORM,
;  255          1242  2         DEL_EXTFCB      : L_NORM NOVALUE,
;  256          1243  2         INIT_FCB2       : L_NORM NOVALUE,
;  257          1244  2         MARK_INCOMPLETE : L_NORM NOVALUE;
;  258          1245  2
;  259          1246  2 IF .BBLOCK [PFCB [FCB$R_ORB], ORB$V_ACL_QUEUE]
;  260          1247  2 THEN
;  261          1248  2     ACL_DELETEACL (PFCB [FCB$L_ACLFL], 0);
;  262          1249  2
;  263          1250  2 DEL_EXTFCB (.PFCB);
;  264          1251  2
;  265          1252  2 IF TESTBITSC (PFCB [FCB$V_STALE])
;  266          1253  2 THEN
;  267          1254  2     IF NOT CONV_ACCLOCK (.PFCB [FCB$B_ACCLKMODE], .PFCB)
;  268          1255  2     THEN
;  269          1256  2         BUG_CHECK (XQPERR, 'unexpected lock manager reaction');
;  270          1257  2
;  271          1258  2 IF .HEADER EQL 0
;  272          1259  2 THEN
;  273          1260  2     RETURN;
;  274          1261  2
;  275          1262  2 INIT_FCB2 (.PFCB, .HEADER);
;  276          1263  2
;  277          1264  2 MARK_INCOMPLETE (.PFCB);
;  278          1265  2
;  279          1266  1 END;                    ! of routine REBLD_PRIM_FCB


                                                .EXTRN  ACL_DELETEACL, CONV_ACCLOCK
                                                .EXTRN  DEL_EXTFCB, MARK_INCOMPLETE
                                                .EXTRN  BUG$_XQPERR

                        0000 00000              .ENTRY  REBLD_PRIM_FCB, Save nothing                    ; 1218
```

```
                          50      04     AC  D0 00002          MOVL    PFCB, R0                                    ; 1246
                10        63      A0     01  E1 00006          BBC     #1, 99(R0), 1$                              ;
                7E        04      AC 00000080   8F  C1 0000D   ADDL3   #128, PFCB, -(SP)                          ; 1248
                        0000G     CF     02  FB 00016          CALLS   #2, ACL_DELETEACL                           ;
                                        04  AC  DD 0001B 1$:    PUSHL   PFCB                                        ; 1250
                        0000G     CF     01  FB 0001E          CALLS   #1, DEL_EXTFCB                              ;
                          50      04     AC  D0 00023          MOVL    PFCB, R0                                    ; 1252
                16        22      A0     08  E5 00027          BBCC    #8, 34(R0), 2$                              ;
                          50      04     AC  D0 0002C          MOVL    PFCB, R0                                    ; 1254
                                        50  DD 00030          PUSHL   R0                                          ;
                          7E      0B     A0  9A 00032          MOVZBL  11(R0), -(SP)                               ;
                        0000G     CF     02  FB 00036          CALLS   #2, CONV_ACCLOCK                            ;
                          04            50  E8 0003B          BLBS    R0, 2$                                      ;
                                      FEFF  0003E              BUGW                                                ; 1256
                                     0000*  00040              .WORD   <BUG$_XQPERR!4>                             ;
                                        08  AC  D5 00042 2$:    TSTL    HEADER                                      ; 1258
                                        11  13 00045          BEQL    3$                                          ;
                          7E      04     AC  7D 00047          MOVQ    PFCB, -(SP)                                 ; 1262
                        0000G     CF     02  FB 0004B          CALLS   #2, INIT_FCB2                               ;
                                        04  AC  DD 00050          PUSHL   PFCB                                        ; 1264
                        0000G     CF     01  FB 00053          CALLS   #1, MARK_INCOMPLETE                         ;
                                            04 00058 3$:        RET                                                 ; 1266

; Routine Size:  89 bytes,    Routine Base:  $CODE$ + 006B

;  280          1267  1
;  281          1268  1 END
;  282          1269  0 ELUDOM


;                                   PSECT SUMMARY
;
;      Name                     Bytes                          Attributes
;
; $CODE$                          196  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         31         0         1000        00:01.9
```

```
;                                       COMMAND QUALIFIERS

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

; Size:         196 code + 0 data bytes
; Run Time:       00:30.8
; Elapsed Time:   00:58.4
; Lines/CPU Min:   2476
; Lexemes/CPU-Min: 67319
; Memory Used:  193 pages
; Compilation Complete
```

DEACCS
LIS

DELETE
LIS

DIRSCN
LIS

DISPAT
LIS

CREHDR
LIS

DIRACC
LIS

DELBAD
LIS

CREFCB
LIS

CREWIN
LIS

DISPATCH
LIS

ENTER
LIS

DELFIL
LIS