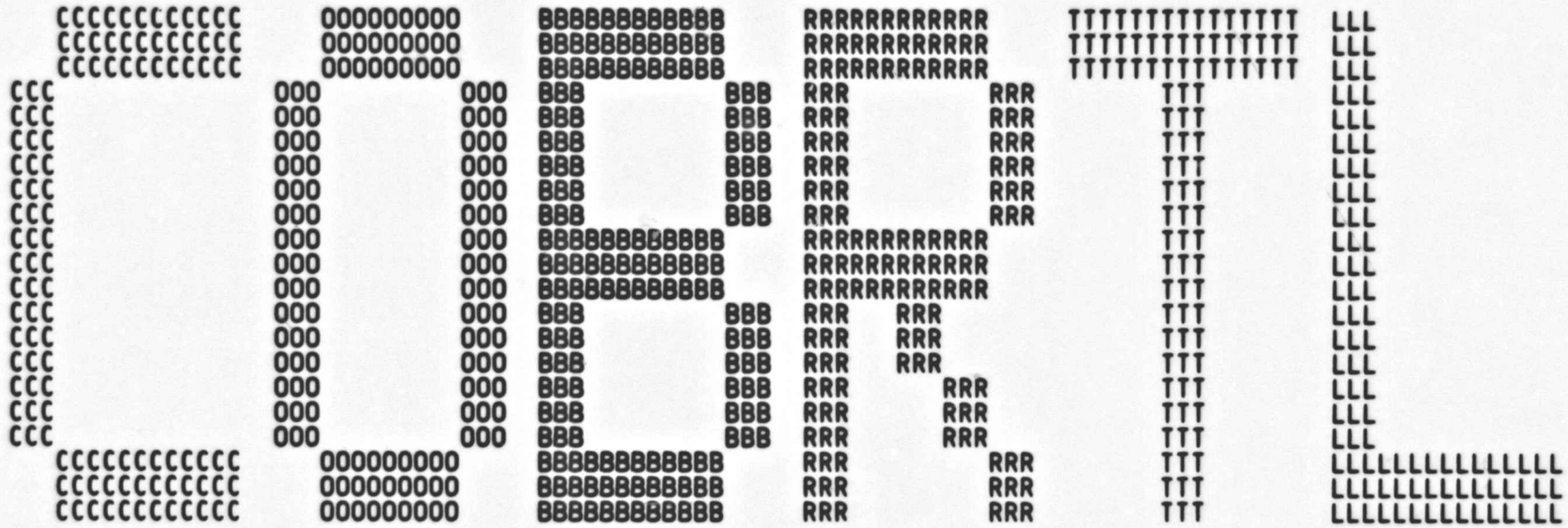
COBRA

```
  CCCCCCCC    OOOOOO    BBBBBBBB     IIIIII    NN      NN  VV      VV  UU      UU    SSSSSSSS  EEEEEEEEEE
  CCCCCCCC    OOOOOO    BBBBBBBB     IIIIII    NN      NN  VV      VV  UU      UU    SSSSSSSS  EEEEEEEEEE
CC           OO      OO  BB      BB     II     NN      NN  VV      VV  UU      UU  SS          EE
CC           OO      OO  BB      BB     II     NN      NN  VV      VV  UU      UU  SS          EE
CC           OO      OO  BB      BB     II     NNNN    NN  VV      VV  UU      UU  SS          EE
CC           OO      OO  BB      BB     II     NNNN    NN  VV      VV  UU      UU  SS          EE
CC           OO      OO  BBBBBBBB       II     NN  NN  NN  VV      VV  UU      UU    SSSSSS    EEEEEEEE
CC           OO      OO  BBBBBBBB       II     NN  NN  NN  VV      VV  UU      UU    SSSSSS    EEEEEEEE
CC           OO      OO  BB      BB     II     NN    NNNN  VV      VV  UU      UU          SS  EE
CC           OO      OO  BB      BB     II     NN    NNNN  VV      VV  UU      UU          SS  EE
CC           OO      OO  BB      BB     II     NN      NN    VV  VV    UU      UU          SS  EE          ....
CC           OO      OO  BB      BB     II     NN      NN    VV  VV    UU      UU          SS  EE          ....
  CCCCCCCC    OOOOOO    BBBBBBBB     IIIIII    NN      NN      VV      UUUUUUUUUU  SSSSSSSS    EEEEEEEEEE  ....
  CCCCCCCC    OOOOOO    BBBBBBBB     IIIIII    NN      NN      VV      UUUUUUUUUU  SSSSSSSS    EEEEEEEEEE  ....

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
```

Table of contents

```
0000     1          .TITLE  COB$$INVOKE_USE COBOL Invoke USE Procedure
0000     2          .IDENT  /1-004/         ; File: COBINVUSE.MAR
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28 ; FACILITY: COBOL SUPPORT
0000    29 ;++
0000    30 ; ABSTRACT:
0000    31 ;       This module contains the routine that invokes a COBOL USE procedure.
0000    32 ;
0000    33 ;
0000    34 ;--
0000    35 ;
0000    36 ; VERSION: 1
0000    37 ;
0000    38 ; HISTORY:
0000    39 ;
0000    40 ; AUTHOR:
0000    41 ;       Marty Jack, 2-May-1979
0000    42 ;
0000    43 ; MODIFIED BY:
0000    44 ;
0000    45 ;
0000    46 ;
```

```
0000     48          .SBTTL  HISTORY          ; Detailed Current Edit History
0000     49
0000     50
0000     51 ; Edit History for Version 1 of COBINVUSE
0000     52 ;
0000     53 ; 1-001 - Original.  MLJ 2-May-1979
0000     54 ; 1-002 - Change entry point name to COB$$INVOKE_USE
0000     55 ; 1-003 - Signal "Recursive activation" directly. Cosmetic changes.
0000     56 ;         RKR 20-OCT-79
0000     57 ; 1-004 - Signalling should be "Recursive activation of USE procedure"
0000     58 ;         RKR 29-OCT-79
```

```
                        60          .SBTTL  DECLARATIONS
                        61
                        62 ;
                        63 ; INCLUDE FILES:
                        64 ;        $SFDEF
                        65 ;
                        66 ;
                        67 ; EXTERNAL SYMBOLS:
                        68          .DISABLE GBL
                        69          .EXTRN  LIB$STOP            ; Signal a condition and stop
                        70          .EXTRN  COB$HANDLER         ; COBOL exception handler
                        71          .EXTRN  COB$_USE_EXIT       ; Special COBOL exception for USE EXIT
                        72          .EXTRN  COB$_RECACTUSE      ; Recursive activation of USE procedure
                        73          .EXTRN  LIB$SIGNAL          ; Signal a condition
                        74          .EXTRN  LIB$STOP            ; Signal a condition and stop
                        75          .EXTRN  OTS$_FATINTERR      ; Fatal internal error
                        76 ;
                        77 ;
                        78 ; MACROS:
                        79 ;        NONE
                        80 ;
                        81 ;
                        82 ;
                        83 ; PSECT DECLARATIONS:
                        84          .PSECT  _COB$CODE           PIC, SHR, LONG, EXE, NOWRT
                        85 ;
                        86 ;
                        87 ; EQUATED SYMBOLS:
                        88 ;
                        89 ;        NONE
                        90 ;
                        91 ; OWN STORAGE:
                        92 ;        NONE
                        93 ;
```

```
                     95          .SBTTL  COB$$INVOKE_USE
                     96
                     97 ;++
                     98 ; FUNCTIONAL DESCRIPTION:
                     99 ;
                    100 ;       This routine invokes a COBOL USE procedure.
                    101 ;
                    102 ; CALLING SEQUENCE:
                    103 ;
                    104 ;       COB$$INVOKE_USE (proc.rl.v, uselist.rl.v, apval.rl.v, eopr.mq.r, pnc.ml.r)
                    105 ;
                    106 ; INPUT PARAMETERS:
                    107 ;
                    108 ;       PROC.rl.v               The address of the USE procedure
                    109 ;       USELIST.rl.v            The address of the USE list of the program
                    110 ;                               in which the original exception was detected
                    111 ;       APVAL.rl.v              The AP of the program in which the original
                    112 ;                               exception was detected
                    113 ;
                    114 ; IMPLICIT INPUTS:
                    115 ;
                    116 ;       NONE
                    117 ;
                    118 ; OUTPUT PARAMETERS:
                    119 ;
                    120 ;       EOPR.mq.r               The end of perform range block for the USE
                    121 ;                               procedure to be invoked.
                    122 ;       PNC.ml.r                The perform nest counter for the module in
                    123 ;                               which the USE procedure is contained.
                    124 ;
                    125 ; IMPLICIT OUTPUTS:
                    126 ;
                    127 ;       NONE
                    128 ;
                    129 ; FUNCTION VALUE:
                    130 ;
                    131 ;       NONE
                    132 ;
                    133 ; SIDE EFFECTS:
                    134 ;
                    135 ;       The USE procedure is executed.
                    136 ;
                    137 ;--
                    138
                    139
 0FFC               140          .ENTRY  COB$$INVOKE_USE,-
0002                141                  ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0002                142                                                  ; All registers MUST be saved since
0002                143                                                  ; the USE procedure will not.
0002                144
00000004            145 proc=   4
00000008            146 uselist=8
0000000C            147 apval=  12
00000010            148 eopr=   16
00000014            149 pnc=    20
0002                150
0002                151 ;+
```

```
                        0002   152 ; Load most parameters into registers.  The USE-list pointer goes to -4(FP)
                        0002   153 ; and the AP value goes to AP.  Note that after this point, the argument
                        0002   154 ; list cannot be accessed, since AP has been modified.  The effect of this
                        0002   155 ; code is that the USE procedure will be executed with the same AP and the
                        0002   156 ; same USE-list as the routine that raised the exception originally.
                        0002   157 ;-
          50   10 AC   D0  0002   158         MOVL    eopr(AP),R0             ; Address of end of perform block
          51   14 AC   D0  0006   159         MOVL    pnc(AP),R1              ; Address of perform nest counter
          52   04 AC   D0  000A   160         MOVL    proc(AP),R2             ; Address of USE procedure
               08 AC   DD  000E   161         PUSHL   uselist(AP)             ; USE-list address to -4(FP)
          5C   0C AC   D0  0011   162         MOVL    apval(AP),AP            ; AP value to AP
                        0015   163
                        0015   164 ;+
                        0015   165 ; Ensure that the USE procedure is inactive.  If it is active, signal an
                        0015   166 ; error.
                        0015   167 ;-
                   60   D5  0015   168         TSTL    (R0)                    ; Test for nonzero return address
                   0D   13  0017   169         BEQL    1$                      ; Br if inactive
         00000000'8F    DD  0019   170         PUSHL   #COB$_RECACTUSE         ; Recursive activation of USE procedure
00000000'GF    01    FB  001F   171         CALLS   #1, G^LIB$STOP          ;
                        0026   172
                        0026   173 ;+
                        0026   174 ; Establish COB$HANDLER as the handler.  This is necessary so that (1) it will
                        0026   175 ; get control if an exception occurs during execution of the USE procedure and
                        0026   176 ; (2) so that COB$IOEXCEPTION will recognize that -4(FP) contains a valid
                        0026   177 ; USE-list pointer in case the USE procedure calls it (recursively).
                        0026   178 ;-
6D   00000000'GF    9E  0026   179 1$:     MOVAB   G^COB$HANDLER,(FP)
                        002D   180
                        002D   181 ;+
                        002D   182 ; Save the return PC from the frame and overwrite it with the address of a
                        002D   183 ; handler that will get control if the USE procedure executes an EXIT PROGRAM
                        002D   184 ; (that is, a RET instruction).
                        002D   185 ;-
               10 AD    DD  002D   186         PUSHL   SF$L_SAVE_PC(FP)        ; Push return PC on stack
     10 AD    46'AF    9E  0030   187         MOVAB   B^10$,SF$L_SAVE_PC(FP)  ; Overwrite with handler address
                        0035   188
                        0035   189 ;+
                        0035   190 ; Invoke the USE procedure using the PERFORM invocation code.
                        0035   191 ;-
                   61   D7  0035   192         DECL    (R1)                    ; Decrement the perform nest counter
          04 A0    61   D0  0037   193         MOVL    (R1),4(R0)              ; Move it to EOPR block
          60    41'AF   9E  003B   194         MOVAB   B^2$,(R0)               ; Move exit address to EOPR block
                   62   17  003F   195         JMP     (R2)                    ; Go to USE procedure
                        0041   196
                        0041   197 ;+
                        0041   198 ; If control reaches this point, the USE procedure terminated by executing
                        0041   199 ; the end of perform range.  Restore the return address, and execute a RET
                        0041   200 ; to return to COB$IOEXCEPTION.
                        0041   201 ;-
     10 AD    8E   D0  0041   202 2$:     MOVL    (SP)+,SF$L_SAVE_PC(FP)  ; Restore return address
                   04   0045   203         RET                             ; Return
                        0046   204
                        0046   205 ;+
                        0046   206 ; If control reaches this point, the USE procedure terminated by executing
                        0046   207 ; an EXIT PROGRAM.  (Note also that this removed the stack frame that we were
                        0046   208 ; executing in, and we are now executing in the frame of COB$IOEXCEPTION.)
```

```
                        0046   209 ; We must cause the program that originally invoked COB$IOEXCEPTION to return.
                        0046   210 ; We do this by signalling a condition that is intercepted by the COB$HANDLER
                        0046   211 ; that is declared in that frame, which will call SYS$UNWIND.
                        0046   212 ;-
      00000000'8F   DD  0046   213 10$:    PUSHL   #COB$_USE_EXIT          ; Special exception condition
00000000'GF   01  FB  004C   214          CALLS   #1, G^LIB$SIGNAL        ; Signal it
      00000000'8F   DD  0053   215          PUSHL   #OTS$_FATINTERR         ; If didn't unwind, fatal error
00000000'GF   01  FB  0059   216          CALLS   #1, G^LIB$STOP          ;
                        0060   217 ;
                        0060   218          .END
```

```
APVAL           =  0000000C
COB$$INVOKE_USE    00000000 RG    02
COB$HANDLER        ********   X   00
COB$_RECACTUSE     ********   X   00
COB$_USE_EXIT      ********   X   00
EOPR            =  00000010
LIB$SIGNAL         ********   X   00
LIB$STOP           ********   X   00
OTS$_FATINTERR     ********   X   00
PNC             =  00000014
PROC            =  00000004
SF$L_SAVE_PC    =  00000010
USELIST         =  00000008
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_COB$CODE	00000060 (96.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.03	00:00:00.96
Command processing	115	00:00:00.32	00:00:02.70
Pass 1	112	00:00:00.69	00:00:05.60
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	54	00:00:00.29	00:00:02.05
Symbol table output	3	00:00:00.01	00:00:00.13
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	318	00:00:01.37	00:00:11.48

The working set limit was 1050 pages.
4095 bytes (8 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 41 non-local and 3 local symbols.
218 source lines were read in Pass 1, producing 13 object records in Pass 2.
8 pages of virtual memory were used to define 7 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	4

88 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

```
MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:COBINVUSE/OBJ=OBJ$:COBINVUSE MSRC$:COBINVUSE/UPDATE=(ENH$:COBINVUSE)
```

COBPAUSE
LIS

COBMSG
LIS

COBHANDLE
LIS

COBINTAR
LIS

COBINTER
LIS

COBMULQ
LIS

COBPOSERA
LIS

COBIOEXCE
LIS

COBLINAGE
LIS

COBKEY
LIS

COBINVUSE
LIS