L 7

```
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
DDD         DDD  CCC              XXX         XXX
DDD         DDD  CCC              XXX         XXX
DDD         DDD  CCC              XXX         XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC                    XXX
DDD         DDD  CCC                    XXX
DDD         DDD  CCC                    XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC                 XXX   XXX
DDD         DDD  CCC              XXX         XXX
DDD         DDD  CCC              XXX         XXX
DDD         DDD  CCC              XXX         XXX
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
DDDDDDDDDDDD        CCCCCCCCCCCC  XXX         XXX
```

```
**FILE**ID**SUBS                                                          M 15
```

```
  SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
  SSSSSSSS  UU      UU  BBBBBBBB      SSSSSSSS
SS          UU      UU  BB      BB  SS
SS          UU      UU  BB      BB  SS
SS          UU      UU  BB      BB  SS
SS          UU      UU  BB      BB  SS
  SSSSSS    UU      UU  BBBBBBBB      SSSSSS
  SSSSSS    UU      UU  BBBBBBBB      SSSSSS
        SS  UU      UU  BB      BB          SS
        SS  UU      UU  BB      BB          SS
        SS  UU      UU  BB      BB          SS      ....
        SS  UU      UU  BB      BB          SS      ....
SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....
SSSSSSSS    UUUUUUUUUU  BBBBBBBB    SSSSSSSS        ....

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
```

```
0 MODULE dcx_subs (                                ! Miscellaneous routines
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-000'
0                       ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 !   FACILITY:
1 !
1 !       DCX -- Data Compression / Expansion Facility
1 !
1 !   ABSTRACT:
1 !
1 !       The Data Compression / Expansion procedures provide a general
1 !       method for reducing the storage requirement for a arbitrary data.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX native, user mode.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  David Thiel
1 !
1 ! CREATION DATE: July, 1981
1 !
1 ! MODIFIED BY:
1 !
1 !--
```

```
;    57          0056   1 %SBTTL  'Declarations';
;    58          0057   1
;    59          0058   1 LIBRARY
;    60          0059   1         'sys$library:starlet';  ! System macros
;    61          0060   1 REQUIRE
;    62          0061   1         'prefix';               ! DCX macro definitions
;    63          0204   1 REQUIRE
;    64          0205   1         'dcxdef';               ! DCX public structure definitions
;    65          0299   1 REQUIRE
;    66          0300   1         'dcxprvdef';            ! DCX private structure definitions
;    67          0466   1
;    68          0467   1 EXTERNAL ROUTINE
;    69          0468   1     lib$free_vm : ADDRESSING_MODE (GENERAL),
;    70          0469   1     lib$get_vm : ADDRESSING_MODE (GENERAL);
;    71          0470   1
;    72          0471   1 EXTERNAL LITERAL
;    73          0472   1     dcx$_invctx,                ! Invalid context block
;    74          0473   1     dcx$_invmap,                ! Invalid map
;    75          0474   1     dcx$_normal;
;    76          0475   1
;    77          0476   1 FORWARD ROUTINE
;    78          0477   1     dcx$get_vm,                 ! Allocate memory
;    79          0478   1     dcx$free_vm,                ! Deallocate memory
;    80          0479   1     dcx$map_check : lkg_map_check,      ! Check map
;    81          0480   1     dcx$ctx_check : lkg_ctx_check,      ! Check context
;    82          0481   1     dcx$long_move : lkg_long_move NOVALUE;      ! Arbitrary length data copy
```

```
1 %SBTTL  'dcx$get_vm -- allocate virtual memory'
1
1 GLOBAL ROUTINE dcx$get_vm (bytes, addr) =
2 BEGIN
2 !++
2 !
2 ! Allocate memory for the data compression / expansion
2 ! facility.  The allocated memory is zeroed.
2 !
2 ! Inputs:
2 !
2 !       bytes                   Number of bytes to allocate
2 !
2 ! Outputs:
2 !
2 !       addr                    Address in which to store addr
2 !                               of allocated memory
2 !
2 ! Return value:
2 !
2 !       dcx$_normal             All is well
2 !       lib$_insvirmem          Error allocation memory
2 !--
2
2 LOCAL
2     p,                      ! pointer for zeroing memory
2     c,                      ! remaining byte count
2     status : LONG;          ! return status
2
2 IF .bytes EQL 0
2 THEN
3     BEGIN
3     .addr = 0;
3     RETURN dcx$_normal;
3     END
3 ELSE IF NOT (status = lib$get_vm (bytes, .addr))
2 THEN
3     BEGIN
3     .addr = 0;
3     RETURN .status;
3     END
2 ELSE
3     BEGIN
3     p = ..addr;
3     c = .bytes;
3     WHILE .c GTRU %X'FFFF' DO
4         BEGIN
4         p = CH$FILL (0, %X'FFFF', .p);
4         c = .c - %X'FFFF';
3         END;
3     CH$FILL (0, .c, .p);
3     RETURN dcx$_normal;
2     END;
1 END;                                      ! of dcx$get_vm


                                                        .TITLE  DCX_SUBS
```

```
                                                                      .IDENT   \V04-000\

                                                                      .EXTRN   LIB$ANALYZE_SDESC_R2
                                                                      .EXTRN   LIB$FREE_VM, LIB$GET_VM
                                                                      .EXTRN   DCX$_INVCTX, DCX$_INVMAP
                                                                      .EXTRN   DCX$_NORMAL

                                                                      .PSECT   $CODE$,NOWRT,2

                                                  007C 00000          .ENTRY   DCX$GET_VM, Save R2,R3,R4,R5,R6                ; 0484
                                          04   AC   D5 00002          TSTL     BYTES                                          ; 0511
                                               05   12 00005          BNEQ     1$                                             ;
                                          08   BC   D4 00007          CLRL     @ADDR                                          ; 0514
                                               3C   11 0000A          BRB      5$                                             ; 0517
                                          08   AC   DD 0000C 1$:      PUSHL    ADDR                                           ;
                                          04   AC   9F 0000F          PUSHAB   BYTES                                          ;
                         00000000G   00        02   FB 00012          CALLS    #2, LIB$GET_VM                                 ;
                                     04        50   E8 00019          BLBS     STATUS, 2$                                     ;
                                          08   BC   D4 0001C          CLRL     @ADDR                                          ; 0520
                                                    04 0001F          RET                                                     ; 0521
                                     53   08   BC   D0 00020 2$:      MOVL     @ADDR, P                                       ; 0525
                                     56   04   AC   D0 00024          MOVL     BYTES, C                                       ; 0526
                         0000FFFF    8F        56   D1 00028 3$:      CMPL     C, #65535                                      ; 0527
                                               11   1B 0002F          BLEQU    4$                                             ;
   FFFF   8F            00           6E        00   2C 00031          MOVC5    #0, (SP), #0, #65535, (P)                      ; 0529
                                               63      00038                                                                  ;
                                     56 FFFF0001  E6   9E 00039       MOVAB    -65535(R6), C                                  ; 0530
                                               E6   11 00040          BRB      3$                                             ; 0527
          56            00           6E        00   2C 00042 4$:      MOVC5    #0, (SP), #0, C, (P)                           ; 0532
                                               63      00047                                                                  ;
                                     50 00000000G 8F  D0 00048 5$:    MOVL     #DCX$_NORMAL, R0                               ; 0533
                                                    04 0004F          RET                                                     ; 0535

; Routine Size:  80 bytes,    Routine Base:  $CODE$ + 0000
;  139          0536  1 %SBTTL  'dcx$free_vm -- free virtual memory'
;  140          0537  1
;  141          0538  1 GLOBAL ROUTINE dcx$free_vm (bytes, addr) =
;  142          0539  2 BEGIN
;  143          0540  2 !++
;  144          0541  2 !
;  145          0542  2 ! Free memory for the data compression / expansion
;  146          0543  2 ! facility.
;  147          0544  2 !
;  148          0545  2 ! Inputs:
;  149          0546  2 !
;  150          0547  2 !       bytes                   Number of bytes to free
;  151          0548  2 !       addr                    Address of block to free
;  152          0549  2 !
;  153          0550  2 ! Outputs:
;  154          0551  2 !
;  155          0552  2 !       NONE
;  156          0553  2 !
;  157          0554  2 ! Return value:
;  158          0555  2 !
;  159          0556  2 !       dcx$_normal             All is well
;  160          0557  2 !       lib$_insvirmem          Error allocating memory
;  161          0558  2 !--
;  162          0559  2
;  163          0560  2 IF .bytes NEQ 0
;  164          0561  2 THEN
;  165          0562  2     perform (lib$free_vm (bytes, addr));
;  166          0563  2 RETURN dcx$_normal;
;  167          0564  2
;  168          0565  1 END;                                    ! of dcx$free_vm



                                  0000 00000          .ENTRY  DCX$FREE_VM, Save nothing                       ; 0538
                       04   AC    D5 00002          TSTL    BYTES                                           ; 0560
                            10    13 00005          BEQL    1$                                              ;
                       08   AC    9F 00007          PUSHAB  ADDR                                            ; 0562
                       04   AC    9F 0000A          PUSHAB  BYTES                                           ;
         00000000G  00            02 FB 0000D          CALLS   #2, LIB$FREE_VM                                 ;
                    07            50 E9 00014          BLBC    STATUS, 2$                                      ;
                    50 00000000G  8F D0 00017 1$:      MOVL    #DCX$_NORMAL, R0                                ; 0563
                                     04 0001E 2$:      RET                                                     ; 0565

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 0050
;  170         0566   1 %SBTTL  'dcx$ctx_check -- check context block'
;  171         0567   1
;  172         0568   1 GLOBAL ROUTINE dcx$ctx_check (ctx : REF BBLOCK, type) : lkg_ctx_check =
;  173         0569   2 BEGIN
;  174         0570   2 !++
;  175         0571   2 !
;  176         0572   2 ! Check context block for validity.
;  177         0573   2 !
;  178         0574   2 ! Inputs:
;  179         0575   2 !
;  180         0576   2 !       ctx                     Address of context block
;  181         0577   2 !       type                    Required context block type
;  182         0578   2 !
;  183         0579   2 ! Outputs:
;  184         0580   2 !
;  185         0581   2 !       NONE
;  186         0582   2 !
;  187         0583   2 ! Return value:
;  188         0584   2 !
;  189         0585   2 !       dcx$_normal             All is well
;  190         0586   2 !       dcx$_invctx             Invalid context block
;  191         0587   2 !       dcx$_invmap             Invalid map
;  192         0588   2 !--
;  193         0589   2
;  194         0590   2 IF .ctx [ctx$l_size] LSSU ctx$k_fixed_len
;  195         0591   2 THEN
;  196         0592   2     RETURN dcx$_invctx
;  197         0593   2 ELSE IF .ctx [ctx$l_sanity] NEQ ctx$c_sanity
;  198         0594   2 THEN
;  199         0595   2     RETURN dcx$_invctx
;  200         0596   2 ELSE IF .ctx [ctx$w_version] NEQ ctx$c_version
;  201         0597   2 THEN
;  202         0598   2     RETURN dcx$_invctx
;  203         0599   2 ELSE IF .ctx [ctx$b_type] NEQ .type
;  204         0600   2 THEN
;  205         0601   2     RETURN dcx$_invctx
;  206         0602   2 ELSE
;  207         0603   2     RETURN dcx$_normal;
;  208         0604   2
;  209         0605   1 END;                                  ! of dcx$ctx_check



                                   14           60  D1 00000 DCX$CTX_CHECK::
                                                                 CMPL     (CTX), #20                          ; 0590
                                                17  1F 00003     BLSSU    1$                                  ;
                        4F317C65   8F       0C  A0  D1 00005     CMPL     12(CTX), #1328643173                ; 0593
                                                0D  12 0000D     BNEQ     1$                                  ;
                                            08  A0  B5 0000F     TSTW     8(CTX)                              ; 0596
                                                08  12 00012     BNEQ     1$                                  ;
         51       04   A0          08           00  ED 00014     CMPZV    #0, #8, 4(CTX), TYPE                ; 0599
                                                08  13 0001A     BEQL     2$                                  ;
                                   50 00000000G 8F  D0 0001C 1$: MOVL     #DCX$_INVCTX, R0                    ; 0601
                                                05     00023     RSB                                          ;
                                   50 00000000G 8F  D0 00024 2$: MOVL     #DCX$_NORMAL, R0                    ; 0603
```

```
                                             05 0002B          RSB                                                  ; 0605

; Routine Size:  44 bytes,    Routine Base:  $CODE$ + 006F
;  211          0606  1 %SBTTL  'dcx$map_check -- check map'
;  212          0607  1
;  213          0608  1 GLOBAL ROUTINE dcx$map_check (dcxmap : REF BBLOCK) : lkg_map_check =
;  214          0609  2 BEGIN
;  215          0610  2 !++
;  216          0611  2 !
;  217          0612  2 ! Check map for validity.
;  218          0613  2 !
;  219          0614  2 ! Inputs:
;  220          0615  2 !
;  221          0616  2 !       dcxmap                  Address of map
;  222          0617  2 !
;  223          0618  2 ! Outputs:
;  224          0619  2 !
;  225          0620  2 !       NONE
;  226          0621  2 !
;  227          0622  2 ! Return value:
;  228          0623  2 !
;  229          0624  2 !       dcx$_normal             All is well
;  230          0625  2 !       dcx$_invmap             Invalid map
;  231          0626  2 !--
;  232          0627  2
;  233          0628  2 IF .dcxmap [dcxmap$l_size] LSSU dcxmap$k_length
;  234          0629  2 THEN
;  235          0630  2     RETURN dcx$_invmap
;  236          0631  2 ELSE IF .dcxmap [dcxmap$l_sanity] NEQ dcxmap$c_sanity
;  237          0632  2 THEN
;  238          0633  2     RETURN dcx$_invmap
;  239          0634  2 ELSE IF .dcxmap [dcxmap$w_version] NEQ dcxmap$c_version
;  240          0635  2 THEN
;  241          0636  2     RETURN dcx$_invmap
;  242          0637  2 ELSE
;  243          0638  2     RETURN dcx$_normal;
;  244          0639  2
;  245          0640  1 END;                                    ! of dcx$map_check



                          14          60  D1 00000 DCX$MAP_CHECK::
                                                       CMPL    (DCXMAP), #20                                   ; 0628
                                      0F  1F 00003     BLSSU   1$                                              ;
                5BF5A3A7  8F      08  A0  D1 00005     CMPL    8(DCXMAP), #1542824871                          ; 0631
                                      05  12 0000D     BNEQ    1$                                              ;
                                  04  A0  B5 0000F     TSTW    4(DCXMAP)                                       ; 0634
                                      08  13 00012     BEQL    2$                                              ;
                          50 00000000G 8F  D0 00014 1$: MOVL    #DCX$_INVMAP, R0                               ; 0636
                                      05  0001B        RSB                                                     ;
                          50 00000000G 8F  D0 0001C 2$: MOVL    #DCX$_NORMAL, R0                               ; 0638
                                      05  00023        RSB                                                     ; 0640

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 009B
;  247          0641  1 %SBTTL  'dcx$long_move -- long data copy'
;  248          0642  1
;  249          0643  1 GLOBAL ROUTINE dcx$long_move (size, source, target) : lkg_long_move NOVALUE =
;  250          0644  2 BEGIN
;  251          0645  2 !++
;  252          0646  2 !
;  253          0647  2 ! Copy arbitrary sized block of data.
;  254          0648  2 !
;  255          0649  2 ! Inputs:
;  256          0650  2 !
;  257          0651  2 !       size                    Length of data to copy
;  258          0652  2 !       source                  Address of data to copy
;  259          0653  2 !       target                  Address of destination
;  260          0654  2 !
;  261          0655  2 ! Outputs:
;  262          0656  2 !
;  263          0657  2 !       NONE
;  264          0658  2 !
;  265          0659  2 ! Return value:
;  266          0660  2 !
;  267          0661  2 !       NONE
;  268          0662  2 !--
;  269          0663  2
;  270          0664  2 BUILTIN
;  271          0665  2     movc3;
;  272          0666  2
;  273          0667  2 WHILE .size GTRU %X'FFFF' DO
;  274          0668  3     BEGIN
;  275          0669  3
;  276          0670  3     LOCAL
;  277          0671  3         dummy0,
;  278          0672  3         dummy2;
;  279          0673  3
;  280          0674  3     movc3 (%REF(%X'FFFF'), .source, .target; dummy0, source, dummy2, target);
;  281          0675  3     size = .size - %X'FFFF';
;  282          0676  2     END;
;  283          0677  2 CH$MOVE (.size, .source, .target);
;  284          0678  2 RETURN;
;  285          0679  2
;  286          0680  1 END;                                    ! of dcx$long_move





                              0078     8F  BB 00000 DCX$LONG_MOVE::
                                                            PUSHR   #^M<R3,R4,R5,R6>                               ; 0643
                    53                 52  D0 00004         MOVL    R2, R3                                         ;
                    56                 50  D0 00007         MOVL    R0, R6                                         ;
       0000FFFF     8F                 56  D1 0000A 1$:     CMPL    SIZE, #65535                                   ; 0667
                                       0F  1B 00011         BLEQU   2$                                             ;
              63    61       FFFF      8F  28 00013         MOVC3   #65535, (SOURCE), (TARGET)                     ; 0674
                    56 FFFF0001        E6  9E 00019         MOVAB   -65535(R6), SIZE                               ; 0675
                                       E8  11 00020         BRB     1$                                             ; 0667
              63    61                 56  28 00022 2$:     MOVC3   SIZE, (SOURCE), (TARGET)                       ; 0677
                              0078     8F  BA 00026         POPR    #^M<R3,R4,R5,R6>                               ; 0680
                                           05 0002A         RSB                                                    ;
```

```
; Routine Size:  43 bytes,    Routine Base:  $CODE$ + 00BF
;  288          0681  1 END                                  ! Of module dcx_subrs
;  289          0682  0 ELUDOM



;                                        PSECT SUMMARY
;
;       Name                         Bytes                          Attributes
;
;  $CODE$                                234  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]STARLET.L32;1                9776         6         0         581           00:01.0





;                                          COMMAND QUALIFIERS

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

; Size:          234 code + 0 data bytes
; Run Time:         00:08.4
; Elapsed Time:     00:33.5
; Lines/CPU Min:     4854
; Lexemes/CPU-Min: 26911
; Memory Used:  76 pages
; Compilation Complete
```

0074 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PREFIX
REQ

STACLINT
LIS

STATUS
LIS

ANALYZE
LIS

DCXMSG
LIS

STASTUB
LIS

DCXDEF
MDL

EXPAND
LIS

SYSOUTPUT
LIS

DCXPRVDEF
MDL

STATEMENT
LIS

DCX

COMPRESS
LIS

TRANSFER
LIS

SYMBOL
LIS

DCXSHR
MAP

SUBS
LIS