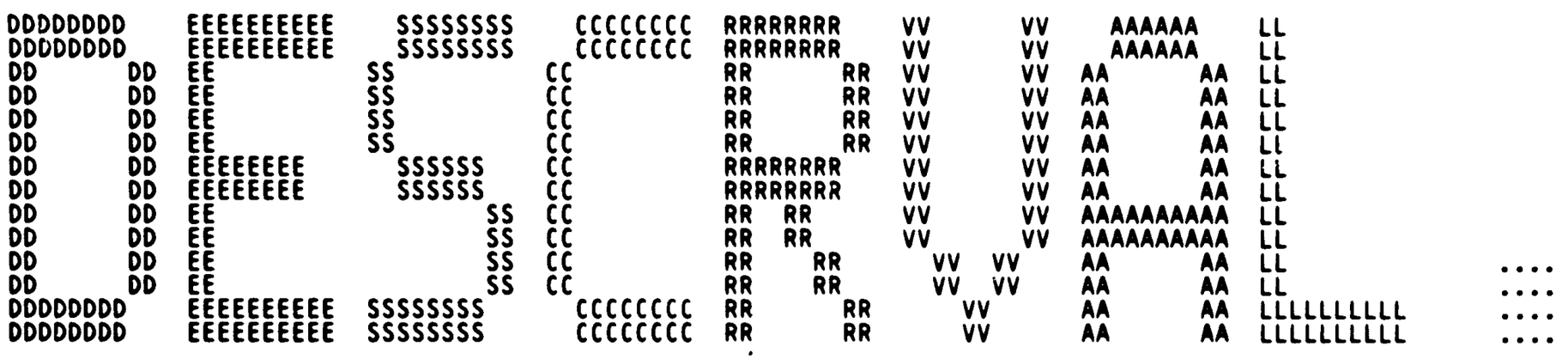
```
DDDDDDDDDDD         CCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
```

FILEID**DESCRVAL



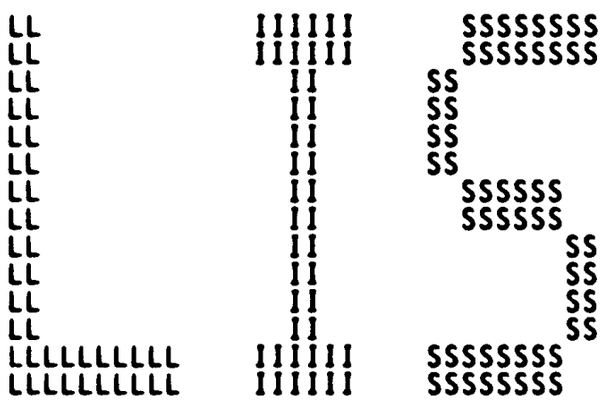


Table of contents

(2) 56 GET RESULT PARSE DESCRIPTOR VALUES
(3) 102 GET KEYWORD NUMBER
(4) 124 SORT THE TOKEN DESCRIPTOR TABLE

```
                        .TITLE  DESCRVAL - TOKEN TABLE MANIPULATION ROUTINES
                        .IDENT  'V04-000'

                ;
                ;*****************************************************************************
                ;*                                                                           *
                ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
                ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                ;*  ALL RIGHTS RESERVED.                                                     *
                ;*                                                                           *
                ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
                ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
                ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
                ;*  TRANSFERRED.                                                             *
                ;*                                                                           *
                ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
                ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
                ;*  CORPORATION.                                                             *
                ;*                                                                           *
                ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
                ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                ;*                                                                           *
                ;*                                                                           *
                ;*****************************************************************************
                ;
                ; D. N. CUTLER 24-APR-77
                ;
                ; PARSE TABLE DESCRIPTOR VALUE MANIPULATION ROUTINES
                ;
                ; MODIFIED BY:
                ;
                ;       V03-004 PCG0004         Peter George            27-May-1983
                ;               Readd DCL$SORT_TOKENS.
                ;
                ;       V03-003 PCG0003         Peter George            06-May-1983
                ;               Move DCL$SORT_TOKENS to PARSENT.
                ;
                ;       V03-002 PCG0002         Peter George            30-Apr-1983
                ;               Add DCL$SORT_TOKENS.
                ;
                ;       V03-001 PCG0001         Peter George            01-Mar-1983
                ;               Add DCL$GETNVAL.
                ;---

                ;
                ; MACRO LIBRARY CALLS
                ;

                        WRKDEF                          ;DEFINE COMMAND WORK AREA
                        PTRDEF                          ;DEFINE TOKEN DESCRIPTOR

00000000                .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
```

```
                     0000     56          .SBTTL  GET RESULT PARSE DESCRIPTOR VALUES
                     0000     57 ;+
                     0000     58 ; DCL$GETDVAL - GET RESULT PARSE DESCRIPTOR VALUES
                     0000     59 ;
                     0000     60 ; THIS ROUTINE IS CALLED TO OBTAIN THE VALUES FROM THE NEXT DESCRIPTOR IN
                     0000     61 ; THE RESULT PARSE TABLE.
                     0000     62 ;
                     0000     63 ; INPUTS:
                     0000     64 ;
                     0000     65 ;       WRK_L_RSLNXT = NEXT TOKEN DESCRIPTOR TO PROCESS
                     0000     66 ;
                     0000     67 ; OUTPUTS:
                     0000     68 ;
                     0000     69 ;       R0 LOW BIT CLEAR INDICATES PARSE DESCRIPTOR TABLE IS EMPTY.
                     0000     70 ;
                     0000     71 ;       R0 LOW BIT SET INDICATES NEXT PARSE DESCRIPTOR HAS BEEN OBTAINED
                     0000     72 ;             WITH:
                     0000     73 ;
                     0000     74 ;       R1 = VALUE FIELD OF DESCRIPTOR.
                     0000     75 ;       R2 = ADDRESS OF ITEM IN COMMAND BUFFER.
                     0000     76 ;       R3 = FLAGS FIELD OF DESCRIPTOR.
                     0000     77 ;       R4 = FIELD TERMINATOR CLASS NUMBER.
                     0000     78 ;       R5 = TYPE OF ENTRY.
                     0000     79 ;-
                     0000     80
                     0000     81 DCL$GETDVAL::                                ;GET RESULT PARSE DESCRIPTOR VALUES
          50   D4   0000     82          CLRL    R0                          ;ASSUME TABLE IS EMPTY
    51    BA AA   D0   0002     83          MOVL    WRK_L_RSLNXT(R10),R1        ;GET ADDRESS OF NEXT TOKEN DESCRIPTOR
 B6 AA    51   D1   0006     84          CMPL    R1,WRK_L_RSLEND(R10)        ;ANY MORE ENTRIES IN TABLE?
          25   1E   000A     85          BGEQU   10$                         ;IF GEQU NO
    BA AA   0C   C0   000C     86          ADDL    #PTR_C_LENGTH,WRK_L_RSLNXT(R10) ;POINT TO NEXT DESCRIPTOR IN TABLE
      0C    08   EF   0010     87          EXTZV   #PTR_V_OFFSET,#PTR_S_OFFSET,- ;EXTRACT OFFSET TO ITEM
      52    61        0013     88                  PTR_L_DESCR(R1),R2          ;
52    F492 CA42   9E   0015     89          MOVAB   WRK_G_BUFFER(R10)[R2],R2    ;CALCULATE ACTUAL ITEM ADDRESS
      04    14   EF   001B     90          EXTZV   #PTR_V_FLAGS,#PTR_S_FLAGS,- ;EXTRACT FLAGS
      53    61        001E     91                  PTR_L_DESCR(R1),R3          ;
      04    18   EF   0020     92          EXTZV   #PTR_V_TERM,#PTR_S_TERM,-   ;EXTRACT TERMINATOR CLASS
      54    61        0023     93                  PTR_L_DESCR(R1),R4          ;
      04    1C   EF   0025     94          EXTZV   #PTR_V_TYPE,#PTR_S_TYPE,-   ;EXTRACT ITEM TYPE
      55    61        0028     95                  PTR_L_DESCR(R1),R5          ;
      08    00   EF   002A     96          EXTZV   #PTR_V_VALUE,#PTR_S_VALUE,- ;EXTRACT VALUE
      51    61        002D     97                  PTR_L_DESCR(R1),R1          ;
            50   D6   002F     98          INCL    R0                          ;INDICATE NEXT DESCRIPTOR OBTAINED
                      0031     99 10$:     RSB                                 ;
                      0032    100
```

```
                     0032   102          .SBTTL  GET KEYWORD NUMBER
                     0032   103 ;+
                     0032   104 ; DCL$GETNVAL - GET KEYWORD NUMBER
                     0032   105 ;
                     0032   106 ; THIS ROUTINE IS CALLED TO OBTAIN THE KEYWORD NUMBER FROM THE CURRENT
                     0032   107 ; DESCRIPTOR IN THE RESULT PARSE TABLE.
                     0032   108 ;
                     0032   109 ; INPUTS:
                     0032   110 ;
                     0032   111 ;       WRK_L_RSLNXT = NEXT TOKEN DESCRIPTOR TO PROCESS
                     0032   112 ;
                     0032   113 ; OUTPUTS:
                     0032   114 ;
                     0032   115 ;       R1 = NUMBER FIELD OF DESCRIPTOR.
                     0032   116 ;-
                     0032   117
                     0032   118 DCL$GETNVAL::                                ; GET KEYWORD NUMBER
          0C   C3    0032   119          SUBL3   #PTR_K_LENGTH,-             ; GET ADDRESS OF TOKEN DESCRIPTOR
     51   BA AA      0034   120                  WRK_L_RSLNXT(R10),R1        ;
     51   05 A1  9A  0037   121          MOVZBL  PTR_B_NUMBER(R1),R1         ; GET KEYWORD NUMBER
                05   003B   122          RSB                                 ;
```

```
                                003C   124          .SBTTL  SORT THE TOKEN DESCRIPTOR TABLE
                                003C   125 ;+
                                003C   126 ; DCL$SORT_TOKENS - SORT THE TOKEN DESCRIPTOR TABLE.
                                003C   127 ;
                                003C   128 ; THIS ROUTINE IS CALLED TO SORT THE TOKEN DESCRIPTOR TABLE INTO
                                003C   129 ; CMDQUAL,PARM,PARMQUAL ORDER TO MAKE GETDVAL PROCESSING IN INTERNAL
                                003C   130 ; CLI ROUTINES EASIER AND TO MAKE DISALLOW PROCESSING EASIER.
                                003C   131 ;
                                003C   132 ; INPUTS:
                                003C   133 ;
                                003C   134 ;       R10 = ADDRESS OF WRK DATA STRUCUTURE
                                003C   135 ;
                                003C   136 ; OUTPUTS:
                                003C   137 ;
                                003C   138 ;       R0-R9 ARE DESTROYED.
                                003C   139 ;
                                003C   140 ;---
                                003C   141 DCL$SORT_TOKENS::
                                003C   142
                          5A    DD   003C   143          PUSHL   R10                     ;SAVE R10
                57  F9B6   CA    9E   003E   144          MOVAB   WRK_G_RESULT(R10),R7    ;GET ADDRESS OF RESULT DESCRIPTOR AR
                    58    57    D0   0043   145          MOVL    R7,R8                   ;COPY ADDRESS OF RESULT DESCRIPTOR A
                5E  FA00  CE    9E   0046   146          MOVAB   -WRK_C_RSLBUFSIZ(SP),SP ;ALLOCATE TEMPORARY DESCRIPTOR ARRAY
                    59    5E    D0   004B   147          MOVL    SP,R9                   ;SAVE ADDRESS OF TEMPORARY DESCRIPTO
                  BA  AA    57    D1   004E   148 10$:     CMPL    R7,WRK_L_RSLNXT(R10)    ;END OF RESULT DESCRIPTOR ARRAY?
                            31    13   0052   149          BEQL    30$                     ;IF EQL YES
                69  67      0C    28   0054   150          MOVC3   #PTR_C_LENGTH,(R7),(R9) ;MOVE DESCRIPTOR TO TEMPORARY ARRAY
                    57      0C    C0   0058   151          ADDL    #PTR_C_LENGTH,R7        ;POINT PAST IT IN ORIGINAL ARRAY
                    04      1C    ED   005B   152          CMPZV   #PTR_V_TYPE,#PTR_S_TYPE,- ;COMMAND QUALIFIER?
                    00      69         005E   153                  (R9),#PTR_K_COMDQUAL     ;
                            EC    12   0060   154          BNEQ    10$                     ;BRANCH IF NOT
          0C    00    6E      00    2C   0062   155          MOVC5   #0,(SP),#0,#PTR_C_LENGTH,- ;CLEAR COMMAND QUALIFIER DESCRIPTOR
                              F4  A7         0067   156                  -PTR_C_LENGTH(R7)  ;  IN ORIGINAL ARRAY
                    59      0C    C0   0069   157          ADDL    #PTR_C_LENGTH,R9        ;POINT TO NEXT ITEM IN TEMPORARY ARR
                    04      1C    ED   006C   158 20$:     CMPZV   #PTR_V_TYPE,#PTR_S_TYPE,- ;QUALIFIER VALUE?
                    02      67         006F   159                  (R7),#PTR_K_QUALVALU     ;
                            DB    12   0071   160          BNEQ    10$                     ;BRANCH IF NOT
                69  67      0C    28   0073   161          MOVC3   #PTR_C_LENGTH,(R7),(R9) ;MOVE DESCRIPTOR TO TEMPORARY ARRAY
                    59      0C    C0   0077   162          ADDL    #PTR_C_LENGTH,R9        ;POINT PAST IT IN TEMPORARY ARRAY
67        0C    00    6E      00    2C   007A   163          MOVC5   #0,(SP),#0,#PTR_C_LENGTH,(R7) ;CLEAR COMMAND QUALIFIER DESCRIPTOR
                    57      0C    C0   0080   164          ADDL    #PTR_C_LENGTH,R7        ;POINT TO NEXT ITEM IN ORIGINAL ARRA
                            E7    11   0083   165          BRB     20$                     ;
                                       0085   166
                    5A      57    D0   0085   167 30$:     MOVL    R7,R10                  ;COPY ADDRESS OF LAST DESCRIPTOR + 1
                    58      57    D1   0088   168 40$:     CMPL    R7,R8                   ;ANY MORE DESCRIPTORS TO EXAMINE?
                            13    13   008B   169          BEQL    50$                     ;IF EQL NO
                    57      0C    C2   008D   170          SUBL    #PTR_C_LENGTH,R7        ;REMOVE DESCRIPTOR
                    56      57    D0   0090   171          MOVL    R7,R6                   ;GET DESCRIPTOR ADDRESS
                            66    D5   0093   172          TSTL    (R6)                    ;WAS THERE A DESCRIPTOR THERE?
                            F1    13   0095   173          BEQL    40$                     ;BRANCH IF NOT
                    5A      0C    C2   0097   174          SUBL    #PTR_C_LENGTH,R10       ;POINT BEFORE IT IN ORIGINAL ARRAY
                6A  66      0C    28   009A   175          MOVC3   #PTR_C_LENGTH,(R6),(R10) ;MOVE DESCRIPTOR TO NEW POSITION
                            E8    11   009E   176          BRB     40$                     ;
                    5E      59    D1   00A0   177 50$:     CMPL    R9,SP                   ;ANY MORE TEMPORARY DESCRIPTORS?
                            0C    13   00A3   178          BEQL    60$                     ;IF EQL NO
                    5A      0C    C2   00A5   179          SUBL    #PTR_C_LENGTH,R10       ;POINT BEFORE IT IN ORIGINAL ARRAY
                    59      0C    C2   00A8   180          SUBL    #PTR_C_LENGTH,R9        ;POINT BEFORE IT IN TEMPORARY ARRAY
```

```
                6A   69    0C    28   00AB   181          MOVC3   #PTR_C_LENGTH,(R9),(R10)        ;MOVE DESCRIPTOR TO NEW POSITION
                           EF    11   00AF   182          BRB     50$                             ;
                                      00B1   183
                5E   0600  CE    9E   00B1   184 60$:     MOVAB   WRK_C_RSLBUFSIZ(SP),SP          ;DEALLOCATE TEMPORARY DESCRIPTOR ARR
                           5A  8ED0   00B6   185          POPL    R10                             ;RESTORE R10
                               05    00B9   186          RSB                                     ;
                                      00BA   187
                                      00BA   188          .END
```

DCL$GETDVAL	00000000	RG	02	WRK_L_TAB_VEC		FFFFFFDE
DCL$GETNVAL	00000032	RG	02	WRK_L_VERB		FFFFFFBE
DCL$SORT_TOKENS	0000003C	RG	02	WRK_W_FLAGS		FFFFFFF0
PTR_B_LEVEL	00000004			WRK_W_FLAGS2		FFFFFFF2
PTR_B_NUMBER	00000005			WRK_W_IMGCHAN		FFFFFFEE
PTR_B_PARMCNT	00000006			WRK_W_PMPTLEN		FFFFF99E
PTR_B_VALUE	00000000					
PTR_C_LENGTH	0000000C					
PTR_K_COMDQUAL	= 00000000					
PTR_K_LENGTH	0000000C					
PTR_K_QUALVALU	= 00000002					
PTR_L_DESCR	00000000					
PTR_L_ENTITY	00000008					
PTR_S_FLAGS	= 00000004					
PTR_S_OFFSET	= 0000000C					
PTR_S_TERM	= 00000004					
PTR_S_TYPE	= 00000004					
PTR_S_VALUE	= 00000008					
PTR_V_FLAGS	= 00000014					
PTR_V_OFFSET	= 00000008					
PTR_V_TERM	= 00000018					
PTR_V_TYPE	= 0000001C					
PTR_V_VALUE	= 00000000					
WRK_B_CMDOPT	FFFFFFC3					
WRK_B_MAXPARM	FFFFFFD0					
WRK_B_MINPARM	FFFFFFD1					
WRK_B_PARMCNT	FFFFFFCE					
WRK_B_PARMSUM	FFFFFFCF					
WRK_B_RECALLCNT	FFFFFFC5					
WRK_B_VALLEV	FFFFFFC4					
WRK_B_VERBTYP	FFFFFFC2					
WRK_C_LENGTH	FFFFF486					
WRK_C_RSLBUFSIZ	= 00000600					
WRK_G_BUFFER	FFFFF492					
WRK_G_INPBUF	FFFFF896					
WRK_G_RESULT	FFFFF9B6					
WRK_K_LENGTH	FFFFF486					
WRK_L_CHARPTR	FFFFF48E					
WRK_L_DISALLOW	FFFFFFE6					
WRK_L_ERRORRTN	FFFFF9AE					
WRK_L_EXPANDPTR	FFFFF486					
WRK_L_IMAGE	FFFFFFE2					
WRK_L_MARKPTR	FFFFF48A					
WRK_L_PAROUT	FFFFFFD2					
WRK_L_PMPTADDR	FFFFF9A2					
WRK_L_PROMPTRTN	FFFFF9A6					
WRK_L_PROPTR	FFFFFFC6					
WRK_L_QUABLK	FFFFFFCA					
WRK_L_READRTN	FFFFF9AA					
WRK_L_RECALLPTR	FFFFFFEA					
WRK_L_RSLEND	FFFFFFB6					
WRK_L_RSLNXT	FFFFFFBA					
WRK_L_SAVAP	FFFFFFF8					
WRK_L_SAVFP	FFFFFFFC					
WRK_L_SAVSP	FFFFFFF4					
WRK_L_SIGNALRTN	FFFFFFD6					
WRK_L_SPECRTN	FFFFF9B2					

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	FFFFFFFC (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
DCL$ZCODE	000000BA (186.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	10	00:00:00.09	00:00:01.44
Command processing	84	00:00:00.61	00:00:07.30
Pass 1	129	00:00:02.26	00:00:07.21
Symbol table sort	0	00:00:00.14	00:00:01.04
Pass 2	38	00:00:00.53	00:00:03.21
Symbol table output	6	00:00:00.06	00:00:00.06
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	269	00:00:03.72	00:00:20.28

The working set limit was 900 pages.
9561 bytes (19 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 122 non-local and 7 local symbols.
188 source lines were read in Pass 1, producing 13 object records in Pass 2.
16 pages of virtual memory were used to define 10 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]SYSBLDMLB.MLB;1	0
_255DUA28:[DCL.OBJ]DCL.MLB;1	2
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	5

207 GETS were required to define 5 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:DESCRVAL/OBJ=OBJ$:DESCRVAL MSRC$:DESCRVAL/UPDATE=(ENH$:DESCRVAL)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

INTDEF
SDL

COMMAND
LIS

DESCRVAL
LIS

CONVERT
LIS

CLIMAC
MAR

CLIMSG
LIS

CONNECT
LIS

DCLPARSE
LIS

CHARMANIP
LIS

EXAMDEP
LIS

EXIT
LIS

INTIMAGES
MAR

DCXSTART
LIS

CLIGBL
LIS

DISALLOW
LIS

CLINT
LIS

CANCEL
LIS