
SSSSSSSSSSS DDDDDDDDDDD AAAAAAAAA
SSSSSSSSSSS DDDDDDDDDDD AAAAAAAAA
SSSSSSSSSSS DDDDDDDDDDD AAAAAAAAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSSSSSSSS DDD DDD AAA AAA
SSSSSSSSS DDD DDD AAA AAA
SSSSSSSSS DDD DDD AAA AAA
SSS DDD DDD AAAAAAAAAAAAAAA
SSS DDD DDD AAAAAAAAAAAAAAA
SSS DDD DDD AAAAAAAAAAAAAAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSS DDD DDD AAA AAA
SSSSSSSSSSS DDDDDDDDDDD AAA AAA
SSSSSSSSSSS DDDDDDDDDDD AAA AAA
SSSSSSSSSSS DDDDDDDDDDD AAA AAA

FILEID**MACROS

```
MM      MM    AAAAAA      CCCCCCCC  RRRRRRRR      000000      SSSSSSSS
MM      MM    AAAAAA      CCCCCCCC  RRRRRRRR      000000      SSSSSSSS
MMMM  MMMM   AA    AA   CC          RR      RR  00    00  SS
MMMM  MMMM   AA    AA   CC          RR      RR  00    00  SS
MM  MM  MM   AA    AA   CC          RR      RR  00    00  SS
MM  MM  MM   AA    AA   CC          RR      RR  00    00  SS
MM      MM   AA    AA   CC          RRRRRRRR    00    00    SSSSSS
MM      MM   AA    AA   CC          RRRRRRRR    00    00    SSSSSS
MM      MM   AAAAAAAAAA   CC          RR  RR      00    00          SS
MM      MM   AAAAAAAAAA   CC          RR  RR      00    00          SS
MM      MM   AA    AA   CC          RR    RR    00    00          SS      ....
MM      MM   AA    AA   CC          RR    RR    00    00          SS      ....
MM      MM   AA    AA     CCCCCCCC  RR      RR    000000    SSSSSSSS      ....
MM      MM   AA    AA     CCCCCCCC  RR      RR    000000    SSSSSSSS      ....
```

```
MM      MM    AAAAAA    RRRRRRRR
MM      MM    AAAAAA    RRRRRRRR
MMMM  MMMM   AA    AA   RR      RR
MMMM  MMMM   AA    AA   RR      RR
MM  MM  MM   AA    AA   RR      RR
MM  MM  MM   AA    AA   RR      RR
MM      MM   AA    AA   RRRRRRRR
MM      MM   AA    AA   RRRRRRRR
MM      MM   AAAAAAAAAA   RR  RR
MM      MM   AAAAAAAAAA   RR  RR
MM      MM   AA    AA   RR    RR
MM      MM   AA    AA   RR    RR
MM      MM   AA    AA   RR      RR
MM      MM   AA    AA   RR      RR
```

```
;
;       MACRO DEFINITIONS FOR SYSTEM DUMP ANALYZER
;
; Version:      '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.                  *
;*                                                                           *
;*****************************************************************************
;
; Author:       Tim Halvorsen
;
; Modified by:
;
;       V03-002 ROW0237         Ralph O. Weber          22-OCT-1983
;               Change uses of .ASCII to .ASCID where appropriate.  Also
;               change to use .ADDRESS where appropriate.  Add ADDR_TABLE
;               macro which is like TABLE but allows caller to specify
;               addresses.  Add PUSHSTR to push the address of a counted
;               ASCII string.  Add COLUMN_LIST, PRINT_COLUMNS, and
;               DO_COLUMN_ENTRY macros to support the print_columns
;               subroutine.  Add MAKE_SYMBOL macro to allow convenient
;               SDA symbol definition.
;
;       V03-001 MSH0001         Maryann Hinden          10-Jun-1982
;               Add PRINTD macro.
;---


;
;       RETIFERR - RETURN IF ERROR CONDITION
;

        .MACRO  RETIFERR ?L1
        BLBS    R0,L1
        RET
L1:
```

```
        .ENDM   RETIFERR


;
;       STATUS - SET RETURN STATUS MESSAGE
;

        .MACRO  STATUS  MSG
        MOVL    #MSG$_'MSG',R0
        .ENDM   STATUS


;
;       SIGNAL - SIGNAL AN ERROR CONDITION
;

        .MACRO  SIGNAL  CNT,MSG,?L1
        .IF     DIF,<CNT>,<RMS>
        .IIF    B,MSG,BLBS      R0,L1
        .IIF    NB,CNT,PUSHL    #CNT
        .IIF    NB,MSG,STATUS   MSG
        PUSHL   R0
        .IIF    NB,CNT,CALLS    #<CNT+2>,G^LIB$SIGNAL
        .IIF    B,CNT,CALLS     #1,G^LIB$SIGNAL
        .IFF
        BLBS    R0,L1
        MOVAB   MSG,R1
        PUSHL   FAB$L_STV(R1)
        PUSHL   R0
        CALLS   #2,G^LIB$SIGNAL
        .ENDC
L1:
        .ENDM   SIGNAL


;
;       ENSURE - ENSURE ENOUGH LINES REMAIN ON THE PAGE
;

        .MACRO  ENSURE  LINECNT,?L1
        SUBL3   LINE_COUNT,PAGE_SIZE,-(SP)
        CMPL    (SP)+,#LINECNT
        BGEQ    L1
        SKIP    PAGE
L1:
        .ENDM   ENSURE


;
;       ERROR - OUTPUT ERROR MESSAGE TO THE TERMINAL
;
;       CNT     = NUMBER OF FAO ARGUMENTS TO PASS TO FAO
```

```
;       TEXT    = FAO CONTROL STRING
;
;
        .MACRO  ERROR,CNT,TEXT,STR
        TYPE    CNT,<TEXT>,STR
        .ENDM   ERROR



;       GETMEM - READ MEMORY, SIGNAL/RETURN IF NOT AVAILABLE
;
;       START   = STARTING VIRTUAL ADDRESS TO READ
;       DEST    = BUFFER TO RETURN MEMORY INTO
;       LENGTH  = LENGTH TO READ
;
        .MACRO  GETMEM START,DEST,LENGTH=#4
        .IF     DIF,LENGTH,#4
        ARGS=3
        PUSHL   LENGTH
        PUSHAL  DEST
        .IFF
        ARGS=1
        .ENDC
        PUSHAB  START
        CALLS   #ARGS,GETMEM
        .IF     IDN,LENGTH,#4
        .IF     NB,DEST
        MOVL    R1,DEST
        .ENDC
        .ENDC
        .ENDM   GETMEM



;       REQMEM - FETCH REQUIRED MEMORY, SIGNAL/EXIT IF NOT AVAILABLE

;       START   = STARTING VIRTUAL ADDRESS TO READ
;       DEST    = BUFFER TO RETURN MEMORY INTO
;       LENGTH  = LENGTH TO READ


        .MACRO  REQMEM START,DEST,LENGTH=#4
        .IF     DIF,LENGTH,#4
        ARGS=3
        PUSHL   LENGTH
        PUSHAL  DEST
        .IFF
        ARGS=1
        .ENDC
        PUSHAB  START
        CALLS   #ARGS,REQMEM
        .IF     IDN,LENGTH,#4
        .IF     NB,DEST
```

```
        MOVL    R1,DEST
        .ENDC
        .ENDC
        .ENDM   REQMEM



;       TRYMEM - FETCH OPTIONAL MEMORY
;
;       START   = STARTING VIRTUAL ADDRESS TO READ
;       DEST    = BUFFER TO RETURN MEMORY INTO
;       LENGTH  = LENGTH TO READ
;

        .MACRO  TRYMEM START,DEST,LENGTH=#4
        .IF     DIF,LENGTH,#4
        ARGS=3
        PUSHL   LENGTH
        PUSHAL  DEST
        .IFF
        ARGS=1
        .ENDC
        PUSHAB  START
        CALLS   #ARGS,TRYMEM
        .IF     IDN,LENGTH,#4
        .IF     NB,DEST
        MOVL    R1,DEST
        .ENDC
        .ENDC
        .ENDM   TRYMEM



;       PUTMEM - STORE INTO DUMP MEMORY, SIGNAL/EXIT IF ERROR
;
;       START   = STARTING DUMP VIRTUAL ADDRESS TO WRITE
;       LOCAL   = LOCAL BUFFER TO TRANSFER FROM
;       LENGTH  = LENGTH TO WRITE
;

        .MACRO  PUTMEM START,LOCAL,LENGTH
        PUSHL   LENGTH
        PUSHAL  LOCAL
        PUSHAB  START
        CALLS   #3,PUTMEM
        .ENDM   PUTMEM



;       PRINT - PRINT A LINE TO THE OUTPUT FILE
;
;       CNT     = NUMBER OF FAO ARGUMENTS TO PASS TO FAO
;       TEXT    = FAO CONTROL STRING
```

```
;
        .MACRO  PRINT,CNT,TEXT,?L1
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
L1:     .ASCID  \TEXT\
        .RESTORE
        PUSHAQ  L1
        CALLS   #<1+CNT>,PRINT
        .ENDM   PRINT



;       PRINTD - print a line to output file, specifying a descriptor
;                for the FAO control string
;
;       CNT  = # FAO ARGS
;       DESC = DESCRIPTOR OF FAO CONTROL STRING
;
        .MACRO  PRINTD,CNT,DESC

        PUSHAQ  DESC
        INCL    CNT
        CALLS   CNT,PRINT

        .ENDM   PRINTD



;       SKIP - SKIP A SPECIFIED NUMBER OF LINES
;
;       CNT     = NUMBER OF BLANK LINES TO OUTPUT


        .MACRO  SKIP,CNT
        .IF     IDN,<CNT>,<PAGE>
        CALLS   #0,NEW_PAGE
        .IFF
        .IF     IDN,<CNT>,<page>
        CALLS   #0,NEW_PAGE
        .IFF
        PUSHL   #'CNT
        CALLS   #1,SKIP_LINES
        .ENDC
        .ENDC
        .ENDM   SKIP



;       STRING - GENERATE A STRING DESCRIPTOR
;
;       STR     = STRING TEXT
```

```
        .MACRO  STRING,STR
        .ASCID  \STR\
        .ENDM   STRING


;
;       PUSHSTR - PUSH THE ADDRESS OF A COUNTED ASCII STRING
;

        .MACRO  PUSHSTR,STR
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
L1:     STRING  <STR>
        .RESTORE
        PUSHAB  L1
        .ENDM   PUSHSTR



;
;       SUBHD - SPECIFY NEW SUB-HEADING FOR LISTING OUTPUT
;
;       TEXT    = TEXT OF NEW SUB-HEADING
;

        .MACRO  SUBHD   TEXT,?L1
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
L1:     STRING  <TEXT>
        .RESTORE
        PUSHAQ  L1
        CALLS   #1,SET_HEADING
        .ENDM   SUBHD



;
;       TYPE - TYPE A SPECIFIED STRING TO THE TERMINAL
;
;       CNT     = NUMBER OF FAO ARGUMENTS TO PASS TO FAO
;       TEXT    = FAO CONTROL STRING
;

        .MACRO  TYPE,CNT,TEXT,STR,?L1
        PUSHAQ  CMND_DESCR
        PUSHAL  OUTPUT+RAB$W_RSZ
        .IF     NB,STR
        PUSHAQ  STR
        .IFF
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
L1:     STRING  <TEXT>
        .RESTORE
        PUSHAQ  L1
        .ENDC
```

```
        CALLS   #<CNT+3>,G^SYS$FAO
        MOVAB   CMND_BUFFER,OUTPUT+RAB$L_RBF
        $PUT    OUTPUT
        SIGNAL  RMS,OUTPUT
        .ENDM   TYPE


;       ALLOC - ALLOCATE BUFFER ON STACK

;       THIS MACRO ALLOCATES SPACE ON THE STACK FOR
;       A BUFFER PRECEEDED BY A DESCRIPTOR DESCRIBING
;       THAT BUFFER.  THE ADDRESS OF THE DESCRIPTOR
;       WILL BE RETURNED IN THE SECOND ARGUMENT.


        .MACRO  ALLOC   LENGTH,RSLDESC
        SUBL    #<LENGTH+3>&<^C3>,SP
        PUSHL   SP
        PUSHL   #LENGTH
        .IF     NB,RSLDESC
        MOVL    SP,RSLDESC
        .ENDC
        .ENDM


;       TABLE - GENERATE DEFINITION TABLE

;       THIS MACRO GENERATES A BIT OR VALUE DEFINITION TABLE
;       FOR USE IN TRANSLATING GIVEN VALUES/BITMASKS INTO THE
;       NAMES CORRESPONDING TO THE VALUES/BITS.


        .MACRO  TABLE,PREFIX,VALUES
        .IRP    VALUE,VALUES
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
$$$     =       .
        .ASCIC  \VALUE\
        .RESTORE
        .LONG   'PREFIX''VALUE'
        .ADDRESS $$$
        .ENDR
        .LONG   -1,-1
        .ENDM   TABLE

;       ADDR_TABLE - GENERATE POINTER TABLE

;       THIS MACRO GENERATES A POINTER DEFINITION TABLE FOR
;       USE IN TRANSLATING GIVEN VALUES INTO THE ADDRESS OF
;       A FURTHER LOOKUP TABLE (OR WHATEVER)


        .MACRO  ADDR_TABLE,PREFIX,LIST
```

```
        .MACRO  $ONE_ADDR$,PFX,VALUE,ADDR
        .LONG   'PFX'-'VALUE'
        .ADDRESS ADDR
        .ENDM   $ONE_ADDR$
        .IRP    ITEM,<LIST>
        $ONE_ADDR$ PREFIX,ITEM
        .ENDR
        .LONG   -1,-1
        .ENDM   ADDR_TABLE


;       PRINT_COLUMNS - GENERATE A CALL TO THE PRINT_COLUMNS ROUTINE
;
;       This macro generates a call to the PRINT_COLUMNS routine.
;
;       Parameters:
;
;         STRUCTURE - address of data structure
;         STRCT_SVA - system virtual address of data structure
;         CLIST1    - COLUMN_LIST base for column 1
;         CLIST2    - COLUMN_LIST base for column 2
;         CLIST3    - COLUMN_LIST base for column 3
;            .              .        .   .    .    .
;            .              .        .   .    .    .
;            .              .        .   .    .    .
;         CLIST10   - COLUMN_LIST base for column 10


        .MACRO  PRINT_COLUMNS,STRUCTURE,STRCT_SVA,CLIST1,CLIST2,CLIST3,CLIST4,-
                        CLIST5,CLIST6,CLIST7,CLIST8,CLIST9,CLIST10
        $$T2 = 2
        .IRP    $$T1,<CLIST10,CLIST9,CLIST8,CLIST7,CLIST6,-
                CLIST5,CLIST4,CLIST3,CLIST2,CLIST1>
        .IF NB $$T1
        PUSHAB  $$T1
        $$T2=$$T2+1
        .ENDC
        .ENDR
        PUSHL   STRCT_SVA
        PUSHAL  STRUCTURE
        CALLS   #$$T2,G^PRINT_COLUMNS
        .ENDM   PRINT_COLUMNS


;       COLUMN_LIST - GENERATE LIST DESCRIBING ONE COLUMN
;                       OF A MULTI-COLUMN TABLE
;
;       This macro generates the description for a single column of
;       a multi-column table.  Each entry in the list has $COLMDEF
;       format.  Such tables are processed by the PRINT_COLUMNS routine.
;       See the PRINT_COLUMNS module header in SDA module MAIN for a
;       description of how use this macro.


        .MACRO  COLUMN_LIST,PREFIX,DFDCSZ,DFVLSZ,DFSPSZ,LIST
        $COLMDEF
```

```
        ASSUME  COLM$K_LENGTH EQ 16
        .MACRO  $COLIST$,PFX,STR,SRC,TYP, -
                DCSZ='DFDCSZ',VLSZ='DFVLSZ',SPSZ='DFSPSZ',?L1
        .SAVE
        .PSECT  LITERALS,EXE,NOWRT
L1:     .ASCIC  \STR\
        .RESTORE
        .ADDRESS L1
        .IF     DF 'PFX''SRC'
        .LONG   -'PFX''SRC'
        .BYTE   COLM$K_FAO_'TYP', 0, 0, 0
        .IFF
        .ADDRESS SRC, TYP
        .ENDC
        .BYTE   DCSZ, VLSZ, SPSZ, 0
        .ENDM   $COLIST$
        .IRP    ITEM,<LIST>
        $COLIST$ PREFIX,ITEM
        .ENDR
        .LONG   0,0,0
        .BYTE   DFDCSZ, DFVLSZ, DFSPSZ, 0
        .ENDM   COLUMN_LIST

;
; DO_COLUMN_ENTRY - USE PRINT_COLUMNS INTERNAL ACTION ROUTINE TO
;                   CONVERT A VALUE
;
; This macro allows a PRINT_COLUMNS action routine to produce one
; of the default output formats using the standard routine.
;
; Parameters:
;
;       type    FAO type (anything valid in the COLUMN_LIST macro is valid here)
;       jump    JMP or JSB controlling transfer to subroutine
;               (JSB is the default)
;
; Inputs:
;
;       R2      address of the datum or its descriptor
;       R5      field size (as input to the action routine)
;

        .MACRO  DO_COLUMN_ENTRY,TYPE,JUMP=JSB
        MOVB    #COLM$K_FAO_'TYPE', R4
        'JUMP'  G^PRINT_COLUMN_VALUE
        .ENDM   DO_COLUMN_ENTRY

;
;       MAKE_SYMBOL - ADD A SYMBOL TO THE SDA SYMBOL TABLE
;


        .MACRO  MAKE_SYMBOL,SYMBOL,VALUE,?L1,?L2
        .SAVE
        .PSECT  LITERALS
L1:     .ASCII  \SYMBOL\
```

```
L2:
        .RESTORE
        PUSHL   VALUE
        PUSHAB  L1
        PUSHL   #<L2-L1>
        CALLS   #3,ADD_SYMBOL
        .ENDM
```

0350 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

XCASE
LIS

SDADEF
SDL

RMSDEF
MAR

UNPUS
LIS

SDATEST
MAP

SDA

SDA
MAP

DCLDEF
MAR

SCSDEF
MAR

TPR
LIS

XTAB
LIS

IPROBE
LIS

SYSDEF
MAR

IMGDEF
MAR

NETDEF
MAR

VAXOPS
REQ

MACROS
MAR

CLUSTER
LIS