

```
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGGGG  RRRRRRRRRRRR  TTTTTTTTTTTTTT  LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMMMMM  MMMMMM  GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSS            MMM      MMM      GGG            RRR      RRR      TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSSSSSSSSSS    MMM      MMM      GGG            RRRRRRRRRRRR  TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGGGGGGGGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSS            SSS      MMM      GGG      GGG      GGG  RRR      RRR      TTT      LLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
SSSSSSSSSSSSS  MMM      MMM      GGGGGGGGGG  RRR      RRR      TTT      LLLLLLLLLLLLLLLL
```

```

SSSSSSSSS MM MM GGGGGGGG DDDDDDDD EEEEEEEEE FFFFFFFF
SSSSSSSSS MM MM GGGGGGGG DDDDDDDD EEEEEEEEE FFFFFFFF
SS MM MMM MMM GG DD DD EE F
SS MM MMM MMM GG DD DD EE F
SS MM MM MM GG DD DD EE F
SS MM MM MM GG DD DD EE F
SSSSSSS MM MM GG GGGGGG DD DD EEEEEEE FFFFFFFF
SSSSSSS MM MM GG GGGGGG DD DD EEEEEEE FFFFFFFF
SS MM MM GG GGGGGG DD DD EE F
SS MM MM GG GGGGGG DD DD EE F
SS MM MM GG GG DD DD EE F
SSSSSSSSS MM MM GGGGGG DDDDDDDD EEEEEEEEE FFFFFFFF
SSSSSSSSS MM MM GGGGGG DDDDDDDD EEEEEEEEE FFFFFFFF

```

```

SSSSSSSSS DDDDDDDD LL
SSSSSSSSS DDDDDDDD LL
SS DD DD LL
SS DD DD LL
SS DD DD LL
SS DD DD LL
SSSSSSS DD DD LL
SSSSSSS DD DD LL
SS DD DD LL
SS DD DD LL
SS DD DD LL
SSSSSSSSS DDDDDDDD LLLLLLLLLL
SSSSSSSSS DDDDDDDD LLLLLLLLLL

```

```
MODULE $SMGDEF;          /* Definitions for RTL Screen Management
{File: SMGDEF.SDL Edit: STAN1017
```

```
{*****
{
{*  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.
{
{*****
```

```
{ Edit history:
```

```
{ 1-001 - Original. SBL 6-April-1983
{ 1-002 - Combine with output version. PLL 14-Jun-1983
{ 1-003 - Remove extra comma on 1st of list to make it compile.
{         RKR 14-JUN-1983
{ 1-004 - Add screen management request type SMG$C_LABEL_BORDER and
{         SMG$C_END_DISPLAY_UPDATE. RKR 20-JUN-1983
{ 1-005 - Add external symbols for terminal types. PLL 21-Jun-1983
{ 1-006 - Add E constants. PLL 18-Aug-1983
{ 1-007 - Add fields to pasteboard info block. STAN 17-Aug-1983.
{ 1-008 - Add DCB truncation icon attribute. PLL 1-Sep-1983
{ 1-009 - Add clear screen on exit attribute. STAN 19-Sep-1983.
{         Add background color wheel.
{ 1-010 - Add DCB display controls attribute. PLL 26-Sep-1983
{ 1-011 - Allow unknown color.
{ 1-012 - Take out FILL from aggregate declaration for the key
{         definition attributes section so that the file generated
{         for C will be correct. LEB 23-Feb-1984
{ 1-013 - Add SMG$V_NOTABS bit for user. STAN 1-Mar-1984.
{ 1-014 - Add literal for start of private terminal types (needed by
{         SHOW TERM). PLL 9-Mar-1984
{ 1-015 - Add SMG$V_KEY_SETSTATE for SMG$GET_KEY_DEFS and SMG$LIST_KEY_DEF
{         of Steve's edit 7 of 26-Aug-1983 that we lost. STAN
{ 1-016 - Need positive longword equivalent of negative byte for private
{         terminal type. PLL 21-Mar-1984
{ 1-017 - Add label positions. STAN 3-Jun-1984.
{
{*
{* Input terminator codes
{*
{* Control characters
```

```
CONSTANT (  
  CTRLA, /* SOH  
  CTRLB, /* STX  
  CTRLC, /* ETX  
  CTRLD, /* EOT  
  CTRL E, /* ENQ  
  CTRL F, /* ACK  
  CTRL G, /* BEL  
  CTRL H, /* BS  
  CTRL I, /* HT  
  CTRL J, /* LF  
  CTRL K, /* VT  
  CTRL L, /* FF  
  CTRL M, /* CR  
  CTRL N, /* SO  
  CTRL O, /* SI  
  CTRL P, /* DLE  
  CTRL Q, /* DC1  
  CTRL R, /* DC2  
  CTRL S, /* DC3  
  CTRL T, /* DC4  
  CTRL U, /* NAK  
  CTRL V, /* SYN  
  CTRL W, /* ETB  
  CTRL X, /* CAN  
  CTRL Y, /* EM  
  CTRL Z, /* SUB  
  ) EQUALS 1 INCREMENT 1 PREFIX SMG$ TAG K_TRM;
```

/* Synonyms for control characters

```
CONSTANT BS EQUALS SMG$K_TRM_CTRLH PREFIX SMG$ TAG K_TRM;  
CONSTANT HT EQUALS SMG$K_TRM_CTRLI PREFIX SMG$ TAG K_TRM;  
CONSTANT LF EQUALS SMG$K_TRM_CTRLJ PREFIX SMG$ TAG K_TRM;  
CONSTANT CR EQUALS SMG$K_TRM_CTRLM PREFIX SMG$ TAG K_TRM;
```

/* Keypad keys

```
CONSTANT (  
  PF1,  
  PF2,  
  PF3,  
  PF4,  
  KP0,  
  KP1,  
  KP2,  
  KP3,  
  KP4,  
  KP5,  
  KP6,  
  KP7,  
  KP8,  
  KP9,  
  ENTER,  
  MINUS,
```

BB
CC
DD
EE
FF
GG
HH
II
JJ
KK
LL
MM
NN
OO
PP
QQ
RR
SS
TT
UU
VV
WW
XX
YY
ZZ

COMMA,
PERIOD,

/* Cursor positioning keys

UP,
DOWN,
LEFT,
RIGHT,
, { 278
, { 279
, { 280

/* Function keys

, { 281
, { 282
, { 283
, { 284
, { 285

F6,

F7,

F8,

F9,

F10,

F11,

F12,

F13,

F14,

HELP, /* F15

DO, /* F16

F17,

F18,

F19,

F20

) EQUALS 256 INCREMENT 1 PREFIX SMG\$ TAG K_TRM;

/* Editing keys

CONSTANT (

FIND,

/* E1

INSERT_HERE,

/* E2

REMOVE,

/* E3

SELECT,

/* E4

PREV_SCREEN,

/* E5

NEXT_SCREEN,

/* E6

) EQUALS 311 INCREMENT 1 PREFIX SMG\$ TAG K_TRM;

/* Synonyms for editing keys

CONSTANT (

E1,

/* FIND

E2,

/* INSERT_HERE

E3,

/* REMOVE

E4,

/* SELECT

E5,

/* PREV_SCREEN

```

E6          /* NEXT SCREEN
) EQUALS 311 INCREMENT 1 PREFIX SMGS TAG K_TRM;

```

```
/* Conditions

```

```

CONSTANT (
  CANCELLED,      /* I/O cancelled by SMG$CANCEL_INPUT
  TIMEOUT,        /* Timeout period expired
  BUFFER_FULL,    /* Buffer is full
  UNKNOWN,        /* Unknown terminator
) EQUALS 508 INCREMENT 1 PREFIX SMGS TAG K_TRM;

```

```

/*
/* Key definition attributes
/*

```

```

AGGREGATE KEY_DEF_ATTR STRUCTURE PREFIX SMGS MARKER SMGS ;
  KEY_NOECHO BITFIELD MASK;      /* Don't echo strings
  KEY_TERMINATE BITFIELD MASK;   /* This key terminates read
  KEY_LOCK BITFIELD MASK;        /* Lock new state
  KEY_PROTECTED BITFIELD MASK;   /* Definition is protected
  KEY_SETSTATE BITFIELD MASK;    /* Definition sets a state (output only)
  dummy1 BITFIELD LENGTH 27 FILL;
END KEY_DEF_ATTR;

```

(Numbers for the following constants begin with the next number after the (SCR\$C_UP_SCROLL constant.

```

constant(                                     /* SCREEN MANAGEMENT REQUEST TYPES
  CHANGE_RENDITION                           /* SMG$CHANGE_RENDITION
  , DELETE_CHARS                             /* SMG$DELETE_CHARS
  , ERASE_DISPLAY                             /* SMG$ERASE_DISPLAY
  , ERASE_LINE                               /* SMG$ERASE_LINE
  , HOME_CURSOR                             /* SMG$HOME_CURSOR
  , INSERT_CHARS                             /* SMG$INSERT_CHARS
  , INSERT_LINE                              /* SMG$INSERT_LINE
  , PUT_CHARS                                /* SMG$PUT_CHARS
  , PUT_LINE                                /* SMG$PUT_LINE
  , PUT_DISPLAY_ENCODED                     /* SMG$PUT_VIRTUAL_DISPLAY_ENCODED
  , RETURN_CURSOR_POS                       /* SMG$RETURN_CURSOR_POS
  , PUT_WITH_SCROLL                         /* SMG$PUT_WITH_SCROLL
  , SET_CURSOR_ABS                          /* SMG$SET_CURSOR_ABS
  , SET_CURSOR_REL                          /* SMG$SET_CURSOR_REL
  , DELETE_LINE                             /* SMG$DELETE_LINE
  , ERASE_CHARS                             /* SMG$ERASE_CHARS
  , SCROLL_DISPLAY_AREA                     /* SMG$SCROLL_DISPLAY_AREA
  , CHANGE_VIRTUAL_DISPLAY                 /* SMG$CHANGE_VIRTUAL_DISPLAY
  , LABEL_BORDER                           /* SMG$LABEL_BORDER
  , END_DISPLAY_UPDATE                     /* SMG$END_DISPLAY_UPDATE
) equals 10 increment 1 prefix SMG tag $C;

```

```

constant(                                     /* Character Set Codes
  UNITED_KINGDOM

```

```

, ASCII
, SPEC_GRAPHICS          /* Special Graphics
, ALT_CHAR                /* Alternate - Standard Char.
, ALT_GRAPHICS            /* Alternate - Special Graphics
) equals 0 increment 1 prefix SMG tag $C;

/*
/* The following constants define corner cursor positions used in
/* SMG$HOME_CURSOR.
/*

constant(
    UPPER_LEFT
    , LOWER_LEFT
    , UPPER_RIGHT
    , LOWER_RIGHT
) equals 0 increment 1 prefix SMG tag $C;

/*
/* The following constants define label positions used in
/* SMG$LABEL_BORDER.
/*

constant(
    TOP
    , BOTTOM
    , LEFT
    , RIGHT
) equals 0 increment 1 prefix SMG;

/*
/* DEFINE BIT MASKS AND VALUES FOR SCREEN ATTRIBUTES
/*

aggregate SMGDEF union prefix SMG$:
    SMGDEF_BITS structure;
        BOLD bitfield mask;          /* Bold rendition
        REVERSE bitfield mask;       /* Reverse video rendition
        BLINK bitfield mask;         /* Blink rendition
        UNDERLINE bitfield mask;     /* Underline rendition
    end SMGDEF_BITS;
    constant "NORMAL" equals 0 prefix SMG$ tag M; /* no bits set

/*
/* DEFINE BITS, MASKS, AND FIELDS FOR THE CONTROL MODES.
/*

SMGMODE_BITS structure;
    BUF_ENABLED bitfield mask; /* Enable buffering
    MINOPD bitfield mask; /* Enable minimal update (default)
    CLEAR_SCREEN bitfield mask; /* Clear screen on exit
    NOTABS bitfield mask; /* Don't use physical tabs
end SMGMODE_BITS;

CONSTANT(    BUF_ENABLED /* Enable Buffering bitvector index
             , MINOPD     /* Enable minimal update bitvector index

```

```

        .CLEAR_SCREEN /* Clear screen on exit
        .NOTABS /* Don't use physical tabs
    ) EQUALS 0 INCREMENT 1;

```

{It would be nice if the above two items could be combined. - stan -

```

/*
/* Master color wheel for screen background colors.
/*

```

```

CONSTANT(UNKNOWN,WHITE,BLACK) EQUALS 0 INCREMENT 1
TAG "C_COLOR";

```

```

/*
/* The following constants correspond to the internal types defined
/* in SMGTERM.REQ. These names used to be for public use - the SMGTERM names
/* used to be needed for compatibility with SCR$ and SMGS code.
/* Use of these names is now very highly discouraged.
/*

```

```

CONSTANT(
    UNKNOWN /* non-graphics or unknown type
    .VT05 /* vt05 series terminal
    .VT52 /* vt52 series terminal
    .VT100 /* vt100 series terminal
    .VTFOREIGN /* foreign terminal (ft1-8)
    .HARDCOPY /* hardcopy device
) EQUALS 0 INCREMENT 1;

```

```

/*
/* The following masks define values to be used
/* to specify a display attribute. These may be added
/* together to specify multiple attributes.
/*

```

{ These must correspond to the DCB_V_xxx attribute bits
{ in the DCB_B_DEF_DISPLAY_ATTR flag byte.

```

DISPLAY_ATTRIBUTES structure;
    BORDER bitfield mask; /* Display is bordered
    TRUNC_ICON bitfield mask; /* Truncation icon enabled if set
    DISPLAY_CONTROLS bitfield mask; /* Display carriage controls if set
end DISPLAY_ATTRIBUTES;

```

```

/*
/* When an out-of-band AST triggers, the user's AST routine
/* will get called. The first argument passed to his
/* routine is the address of a Band Information Table.
/* The offsets into this table are described below.
/* To make it easy to access from certain languages,
/* the table is arranged so that it is convenient to
/* reference it as a longword vector as well as a byte block.
/*

```

```

BAND_INFORMATION_TABLE structure;
    PASTEBOARD_ID longword unsigned; /* Pasteboard ID
    ARG longword unsigned; /* User's AST argument

```

```

CHARACTER OVERLAY union;
  "CHARACTER" byte unsigned; /* The character typed
  "CHARACTER" longword unsigned; /* The character and 3 spaces
end CHARACTER_OVERLAY;
end BAND_INFORMATION_TABLE;

```

```

/* When SMG$GET_PASTEBOARD_ATTRIBUTES is called,
/* it returns data in a pasteboard information block.
/* The following structure is used to reference fields
/* in this block.
/* Items marked with a plus (+) will be 0 unless the
/* device is a terminal (DEVCLASS=DC$_TERM).

```

```

PASTEBOARD_INFO_BLOCK structure;
  DEVCHAR longword unsigned; /* Device characteristics
  DEVDEPEND longword unsigned; /* Specific characteristics (1)
  DEVDEPEND2 longword unsigned; /* Specific characteristics (2)
  DEVCLASS byte unsigned; /* Device class (e.g. DC$_TERM)
  SMG_DEVTTYPE byte unsigned; /* Internal SMG device type
  PHY_DEVTTYPE byte unsigned; /* Physical device type (e.g. DT$_VT100)
  ROWS byte unsigned; /* Number of rows on device
  WIDTH word unsigned; /* Terminal width
  COLOR byte unsigned; /* Reserved for future use
  PARITY byte unsigned; /* Parity attributes (+)
  SPEED word unsigned; /* Terminal Speed (+)
  "FILL" word unsigned; /* Fill characteristics (+)
  CURSOR_ROW word unsigned; /* Row where physical cursor is
/* (1-origin)
  CURSOR_COL word unsigned; /* Col where physical cursor is
/* (1-origin)
  CURSOR_DID longword unsigned; /* Display id of topmost
/* display that cursor is in.
/* 0 if cursor is not in a
/* virtual display

```

```

end PASTEBOARD_INFO_BLOCK;

```

```

/*
/* Directions are set up as a structure, assuming that directions
/* may be combined in the future (ie. diagonal movement).
/*

```

```

SCROLL_DIRS structure;
  UP bitfield mask; /* Scroll up
  DOWN bitfield mask; /* Scroll down
  RIGHT bitfield mask; /* Scroll right
  LEFT bitfield mask; /* Scroll left
end SCROLL_DIRS;

```

```

( Foreign types are in the range 128 - 255 (or negative numbers -1 to -127,
( since type is stored as a byte). Half of the types are reserved for private
( termtables (-65 to -127), and half are reserved for the system termtable
( (-1 to -64). Provide a constant for use by SHOW TERM so that it doesn't
( need to know the distinction.

```

```

CONSTANT FIRST_PRIV_TYPE EQUALS 191 PREFIX SMG$;

```

end SMGDEF;

end_module \$SMGDEF;

0354 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYMBOLS
LIS

SMGRTL

SMGBLDTM
MAP

SDAMSG
LIS

VAXINST
LIS

SMGMAPTRM
MAP

SMGKCB
SDL

VALIDATE
LIS

STACKS
LIS

SMGDEF
SDL

SMGKDE
SDL

SMGSHR
MAP