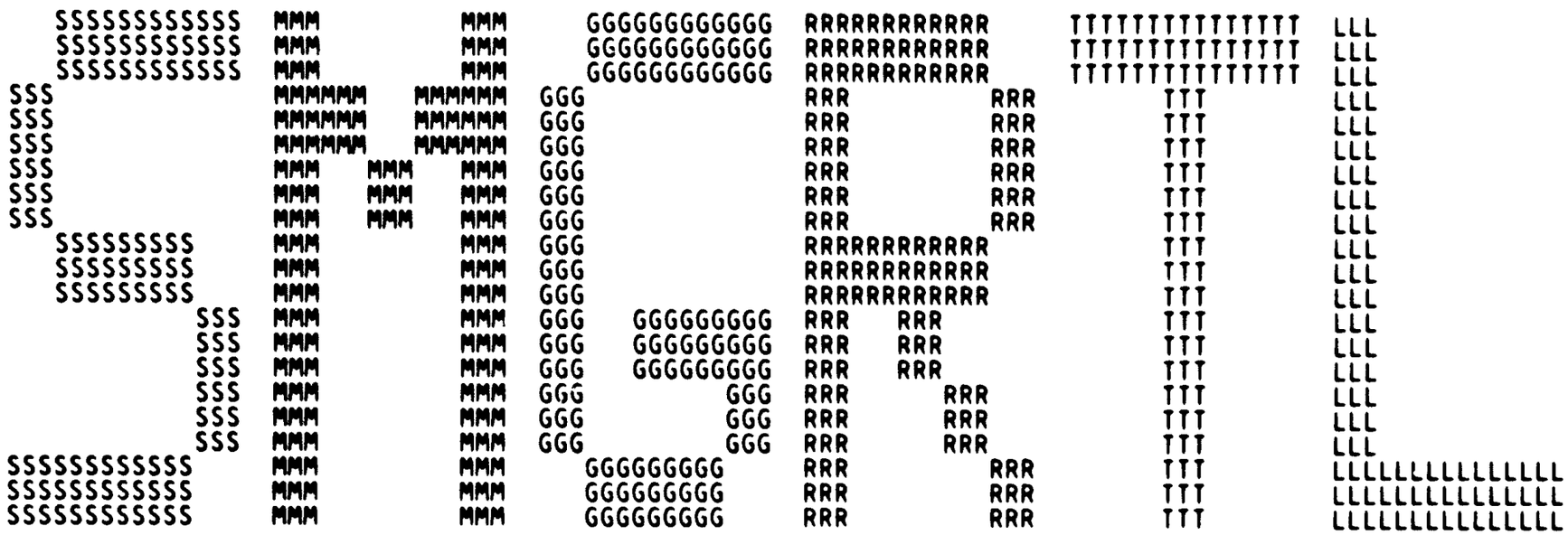


_$2

Val

001
001
001
001
001
001
001
001
001
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF

```
  SSSSSSSS  MM      MM    GGGGGGGG  TTTTTTTTTT    AAAAAA    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
  SSSSSSSS  MM      MM    GGGGGGGG  TTTTTTTTTT    AAAAAA    BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
SS          MMMM  MMMM  GG              TT      AA      AA  BB      BB  DD      DD  EE          FF
SS          MMMM  MMMM  GG              TT      AA      AA  BB      BB  DD      DD  EE          FF
SS          MM  MM  MM  GG              TT      AA      AA  BB      BB  DD      DD  EE          FF
SS          MM  MM  MM  GG              TT      AA      AA  BB      BB  DD      DD  EE          FF
  SSSSSS    MM      MM  GG              TT      AA      AA  BBBBBBBB    DD      DD  EEEEEEEE    FFFFFFFF
  SSSSSS    MM      MM  GG              TT      AA      AA  BBBBBBBB    DD      DD  EEEEEEEE    FFFFFFFF
        SS  MM      MM  GG  GGGGGG      TT      AAAAAAAAAA  BB      BB  DD      DD  EE          FF
        SS  MM      MM  GG  GGGGGG      TT      AAAAAAAAAA  BB      BB  DD      DD  EE          FF
        SS  MM      MM  GG      GG      TT      AA      AA  BB      BB  DD      DD  EE          FF                ....
        SS  MM      MM  GG      GG      TT      AA      AA  BB      BB  DD      DD  EE          FF                ....
SSSSSSSS    MM      MM    GGGGGG        TT      AA      AA  BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FF                ....
SSSSSSSS    MM      MM    GGGGGG        TT      AA      AA  BBBBBBBB    DDDDDDDD    EEEEEEEEEE  FF                ....
```

```
RRRRRRRR    EEEEEEEEEE    QQQQQQ
RRRRRRRR    EEEEEEEEEE    QQQQQQ
RR      RR  EE          QQ      QQ
RR      RR  EE          QQ      QQ
RR      RR  EE          QQ      QQ
RR      RR  EE          QQ      QQ
RRRRRRRR    EEEEEEEE    QQ      QQ
RRRRRRRR    EEEEEEEE    QQ      QQ
RR  RR      EE          QQ  QQ  QQ
RR  RR      EE          QQ  QQ  QQ
RR    RR    EE          QQ    QQ
RR    RR    EE          QQ    QQ
RR      RR  EEEEEEEEEE    QQQQ  QQ
RR      RR  EEEEEEEEEE    QQQQ  QQ
```

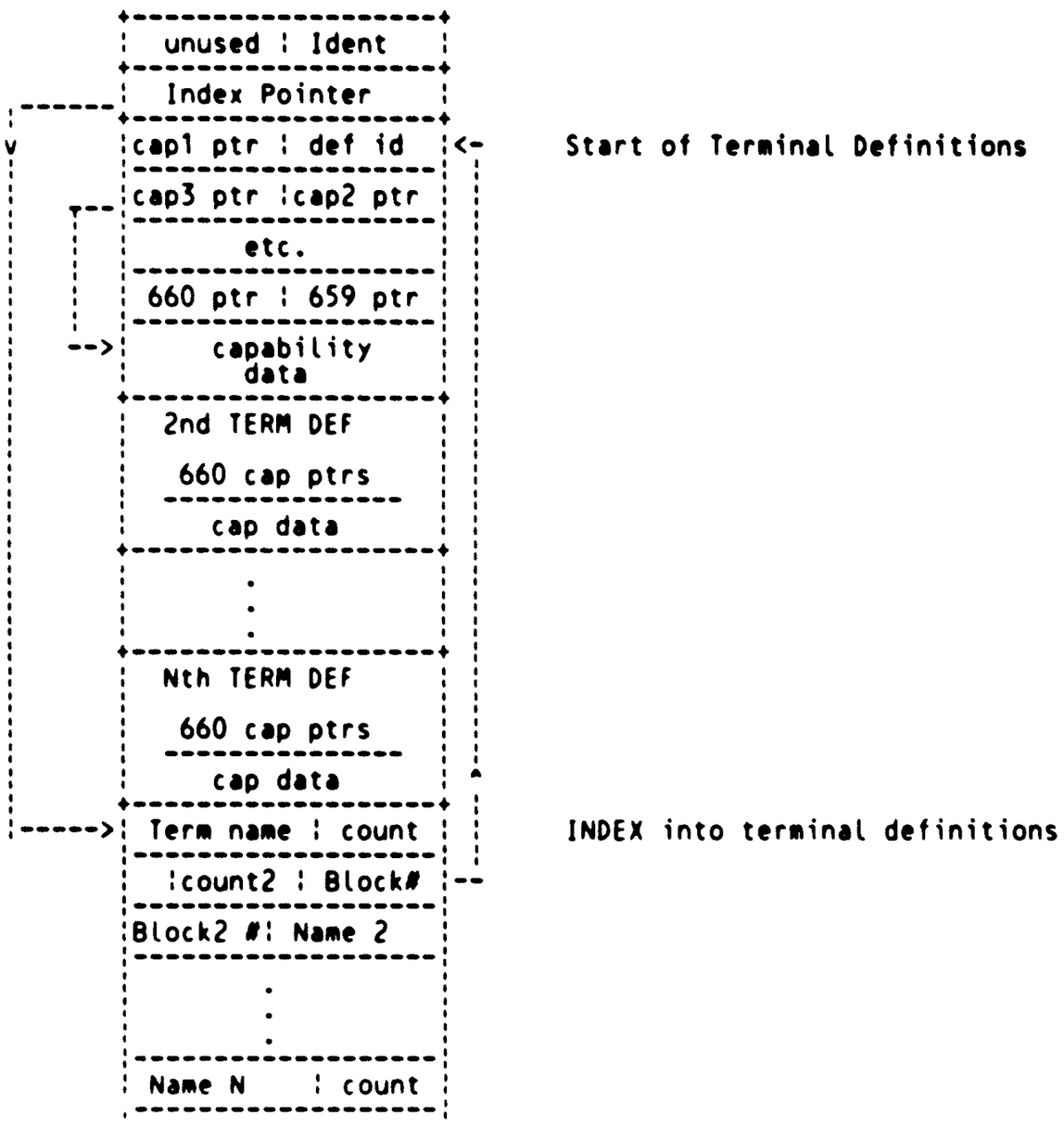
```
! Terminal Table Data Structure Definitions for RTL SMG$ facility
! File:  SMGTABDEF.REQ  Edit: PLL1001
!
```



```
!++
! FACILITY:     Screen Management
!
! ABSTRACT:
!
!       This module defines the internal structure of the binary
!       TERMTABLE.EXE.
!
!
! MODIFIED BY:
!
! 1-001 - Original.  PLL 2-Nov-1983
!--
!
! The internal structure of TERMTABLE.EXE is divided into several sections.
!
! Each terminal definition consists of 660 capability pointers and
! a data area.  If a terminal does not have a particular capability defined,
! the pointer for that capability will be zero.  Otherwise, the capability
! pointer is an offset into the data area.
!
! Each capability data consists of a count and the data.  The data may be
! a binary number or an ascii string.  A negative count indicates that
! this is a dynamic string which requires processing (substitution, conversion,
! or arithmetic).
```

TERMTABLE.EXE also contains an index so that individual terminal definitions can be located. The index will consist of a count, the ascii name for the terminal, and the block number where the terminal definition begins. The index will actually be located at the end of TERMTABLE.EXE. This is because the number of terminal definitions is unpredictable, and we don't want to impose an arbitrary size constraint that would restrict the number of terminals defined.

TERMTABLE.EXE will begin with an ident number, to allow for future changes. These future changes would probably allow more terminal capabilities (greater than 660).



```
!                 !  !   0   ! BlockN#! count = 0 --> end of terminal def ptrs
!                 +-------------------+
!
!+
! Offsets used to access a terminal table.
!-

MACRO
        TTB_W_IDENT             = 0, 0, 16, 0%, ! ident field (to allow for
                                                ! future changes)
        TTB_W_unused            = 2, 0, 16, 0%, ! not used

        TTB_L_INDEX_OFFSET      = 4, 0, 32, 0%; ! offset from here to terminal
                                                ! index

!+
! Define constants needed to build terminal table
!-

LITERAL
        SMG$K_HEADER_SIZE       = 512,          ! 1 block for header info
        SMG$K_TERM_INDEX_SIZE   = 5000,         ! arbitrary size for index
        SMG$K_CAP_PTRS_SIZE     = 1536,         ! 3 blocks to hold ptrs
        SMG$K_CAP_DATA_SIZE     = 5120,         ! 10 blocks to hold data
        SMG$K_TERM_DEF_SIZE     = 6656,         ! ptrs + data (3+10 blocks)
        SMG$K_TERM_DEF_ID       = 137,          ! random number to identify
                                                ! a valid terminal def
        SMG$K_TERM_DEF_ID_OFFSET = 0,           ! location of id - 1st
                                                ! word in ptr vector
        SMG$K_MIN_REQUEST_CODE  = 1,            ! lowest capability number
        SMG$K_MAX_REQUEST_CODE  = 660;          ! highest capability number

!+
! The following are used when parsing arithmetic expressions.
! Information about expressions is stored via these encodings.
!-
LITERAL
        SMG$K_FAO_STRING        = %X'FFFFFFFF',
        SMG$K_ARITH_STRING      = %X'FFFFFFFE',
        SMG$K_OPERAND           = %X'FFFFFFFD',
        SMG$K_SUBSTITUTE        = %X'FFFFFFFC',
        SMG$K_ADD               = %X'FFFFFFFB',
        SMG$K_SUBTRACT          = %X'FFFFFFFA',
        SMG$K_MULTIPLY          = %X'FFFFFFF9',
        SMG$K_DIVIDE            = %X'FFFFFFF8',
        SMG$K_STORE             = %X'FFFFFFF7';
```

SMGDATSTR
REQ

SMGLNK
REQ

SMGKQB
SDL

SMGBLDTRM
LIS

SMGDISCHA
LIS

SMGKTH
SDL

SMGMACROS
REQ

SMGPROLOG
REQ

SMGTERM
REQ

SMGTRMSTR
R32

SMGTRMPTR
SDL

SMGSCRMAC
REQ

SMGALLESC
LIS

SMGTPACTL
REQ

SMGSCRTCB
REQ

SMGLIB
REQ

SMGTPALIB
REQ

SMGDEFKEY
LIS

SMGTABDEF
REQ

SMGTRMMAC
REQ

SMGBOOTAB
LIS