```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS  FFFFFFFFFF  RRRRRRRR      AAAAAA    MM      MM  EEEEEEEEEE
BBBBBBBB      AAAAAA      SSSSSSSS  FFFFFFFFFF  RRRRRRRR      AAAAAA    MM      MM  EEEEEEEEEE
BB      BB   AA      AA  SS        FF          RR      RR   AA      AA  MMMM  MMMM  EE
BB      BB   AA      AA  SS        FF          RR      RR   AA      AA  MMMM  MMMM  EE
BB      BB   AA      AA  SS        FF          RR      RR   AA      AA  MM  MM  MM  EE
BB      BB   AA      AA  SS        FF          RR      RR   AA      AA  MM  MM  MM  EE
BBBBBBBB     AA      AA    SSSSSS  FFFFFFFF    RRRRRRRR     AA      AA  MM      MM  EEEEEEEE
BBBBBBBB     AA      AA    SSSSSS  FFFFFFFF    RRRRRRRR     AA      AA  MM      MM  EEEEEEEE
BB      BB   AAAAAAAAAA          SS  FF          RR  RR       AAAAAAAAAA  MM      MM  EE
BB      BB   AAAAAAAAAA          SS  FF          RR  RR       AAAAAAAAAA  MM      MM  EE
BB      BB   AA      AA          SS  FF          RR    RR     AA      AA  MM      MM  EE
BB      BB   AA      AA          SS  FF          RR    RR     AA      AA  MM      MM  EE
BBBBBBBB     AA      AA  SSSSSSSS    FF          RR      RR   AA      AA  MM      MM  EEEEEEEEEE
BBBBBBBB     AA      AA  SSSSSSSS    FF          RR      RR   AA      AA  MM      MM  EEEEEEEEEE

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
```

```
!+
! This file, BASFRAME.REQ, defines the frame control data for
! a BASIC procedure.            Edit: PLL1010
!-

!


!+
! Edit History:
!
! 0-001 Initial coding from BP2VAXDGC.  JBS 19-NOV-78
! 1-001 - Make version 1 to conform to version numbering standard.
!         (The conversion from BAS$ prefixes to BSF$ prefixes will
!         be in a future revision.)  JBS 27-NOV-78
! 1-002 - Convert from BAS to BSF prefixes, and add BSF$A_BASE_R9.
!         JBS 08-FEB-1979
! 1-003 - Add BSF$K_LENFCDMAJ, BSF$K_LENFCDDEF and BSF$K_LENFCDDFS.
!         JBS 09-FEB-1979
! 1-004 - Add BSF$V_FCD_IV.  JBS 11-SEP-1979
! 1-005 - Remove the PRINT statement, for the new BLISS compiler.
!         JBS 02-OCT-1979
! 1-006 - Add macro to pick up scale.  29-Oct-79
! 1-007 - Add copyright notice.  SBL 11-Mar-1980
! 1-008 - Add BSF$A_RTA_DESC and BSF$M_FCD_DV.  PLL 12-May-1982
! 1-009 - Add BSF$M_FCD_RND.  PLL 10-Jun-1982
! 1-010 - Change def of BSF$A_RTA_DESC (was overlapping BSF$A_USER_HAND).
!         PLL 10-Aug-1982
!--

FIELD
    BSF$FCD =
        SET
!+
! The following appear only in major frames.
```

```
!+
        BSF$A_RTA_DESC = [-72, 0, %BPVAL, 0],   ! ptr to run-time array dscs
!       BSF$A_USER_HAND = [-68, 0, %BPVAL, 0],  ! (never accessed from FP)
        BSF$A_BASE_PC = [-64, 0, %BPVAL, 0],    ! first byte of code
        BSF$A_CUR_DATA = [-60, 0, %BPVAL, 0],   ! current byte of data text
        BSF$A_END_DATA = [-56, 0, %BPVAL, 0],   ! last byte + 1 of data text
        BSF$B_SCA_V_PAC = [-52, 0, 8, 1],       ! scale for packed, 0 to -6.
        BSF$B_SCA_V_DOU = [-51, 0, 8, 1],       ! scale for double, 0 to -6.
        BSF$D_SCALE_DOU = [-48, 0, 0, 0],       ! scale for double, 1 to 10**6
!+
! The following are also in DEF and DEF* frames
!-
        BSF$A_INIT_ARG = [-40, 0, %BPVAL, 0],   ! pointer to INIT arg list
        BSF$L_INIT_REL = [-36, 0, %BPVAL, 1],   ! relocation for INIT arg list
        BSF$A_STR_DESC = [-32, 0, %BPVAL, 0],   ! dynamic string descriptors
!+
! The following are also in GOSUB, CONDITION HANDLING and IOL frames
!+
        BSF$B_LEN_FCD = [-28, 0, 8, 0],         ! length of FCD
        BSF$B_PROC_CODE = [-27, 0, 8, 1],       ! frame type, see below
        BSF$W_FCD_FLAGS = [-26, 0, 16, 0],      ! frame flags, see below
        BSF$A_PROC_ID = [-24, 0, %BPVAL, 0],    ! info for frame
        BSF$A_BASE_R9 = [-20, 0, %BPVAL, 0],    ! R9 for this procedure
        BSF$A_BASE_R10 = [-16, 0, %BPVAL, 0],   ! R10 for this procedure
        BSF$A_BASE_R11 = [-12, 0, %BPVAL, 0],   ! R11 for this procedure
        BSF$A_BASE_SP = [-8, 0, %BPVAL, 0],     ! SP for this procedure
        BSF$A_MARK = [-4, 0, %BPVAL, 0],        ! last mark PC
!+
! The following are also in non-BASIC frames.
!-
        BSF$A_HANDLER = [0, 0, %BPVAL, 0],      ! exception handler
        BSF$A_SAVED_AP = [8, 0, %BPVAL, 0],     ! previous value of AP
        BSF$A_SAVED_FP = [12, 0, %BPVAL, 0],    ! previous value of FP
        BSF$A_SAVED_PC = [16, 0, %BPVAL, 0]     ! return address in previous frame
        TES;

!+
! Define the frame type codes.
!-

LITERAL
    BSF$K_PROC_MAIN = 1,                        ! main program
    BSF$K_PROC_SUB = 2,                         ! subprogram
    BSF$K_PROC_EXTF = 3,                        ! external function
    BSF$K_PROC_DEF = 4,                         ! DEF function
    BSF$K_PROC_DEFS = 5,                        ! DEF* function
    BSF$K_PROC_GOSB = 6,                        ! GOSUB (subroutine)
    BSF$K_PROC_ONER = 7,                        ! condition handler
    BSF$K_PROC_IOL = 8;                         ! Immediate, On-Line

!+
! Define the bits in the flags word.
!-

LITERAL
    BSF$M_FCD_LONG = 1^15,                      ! compiled with 32-bit integers
```

```
    BSF$M_FCD_DOU = 1^14,                   ! compiled with double floating
    BSF$M_FCD_RSTR = 1^13,                  ! procedure returns a string result
    BSF$M_FCD_OEGO = 1^12,                  ! special ON ERROR initial processing
    BSF$M_FCD_IV = 1^11,                    ! this procedure has IV set
    BSF$M_FCD_DV = 1^10,                    ! decimal overflow enabled if set
    BSF$M_FCD_RND = 1^9;                    ! decimal rounding if set
!+
! The frame control data for the currently active major procedure
! is pointed to by R11, but R11 is offset by 195 bytes from the
! base of the frame so that the compiled code can more frequently
! use byte offsets from R11 to address its local variables.
! Therefore, to address the frame of the major procedure we need
! some new names.  To avoid a proliferation of names, we name below
! only the fields we reference directly off of R11; the other fields
! are referenced by removing the offset first.
!-

FIELD
    BSF$MAJOR_FRAME =
        SET
        BSF$A_USER_HAND = [127, 0, %BPVAL, 0],  ! user's error handling flag
        BSF$A_CODE_BEG = [131, 0, %BPVAL, 0],   ! first byte of code
        BSF$A_CUR_DTA = [135, 0, %BPVAL, 0],    ! current byte of DATA text
        BSF$A_END_DTA = [139, 0, %BPVAL, 0],    ! last byte + 1 of DATA text
        BSF$B_SCA_V_PAC = [143, 0, 8, 1],       ! scale for packed, 0 to -6.
        BSF$B_SCA_V_DOU = [144, 0, 8, 1],       ! scale for double, 0 to -6.
        BSF$D_SCALE_DOU = [147, 0, 0, 0],       ! scale for double, 1 to 10**6
        BSF$A_INIT_ARG = [155, 0, %BPVAL, 0],   ! pointer to INIT arg list
        BSF$L_INIT_REL = [159, 0, %BPVAL, 1],   ! relocation for INIT arg list
        BSF$A_STR_DESC = [163, 0, %BPVAL, 0],   ! dynamic string descriptors
        BSF$B_LEN_FCD = [167, 0, 8, 0],         ! length of FCD
        BSF$B_PROC_CODE = [168, 0, 8, 1],       ! frame type, see above
        BSF$W_FCD_FLAGS = [169, 0, 16, 0],      ! frame flags, see below
        BSF$A_PROC_INFO = [171, 0, %BPVAL, 0],  ! info for frame, see BSF$A_PROC_ID
        BSF$A_BASE_R9 = [175, 0, %BPVAL, 0],    ! R9 for this procedure
        BSF$A_BASE_R10 = [179, 0, %BPVAL, 0],   ! R10 for this procedure
        BSF$A_BASE_R11 = [183, 0, %BPVAL, 0],   ! R11 for this procedure
        BSF$A_BASE_SP = [187, 0, %BPVAL, 0],    ! SP for this procedure
        BSF$A_MARK = [191, 0, %BPVAL, 0],               ! last mark PC
        BSF$FRAME_BASE = [195, 0, 0, 0]         ! base of major frame
        TES;

!+
! The minor frame is arranged differently because the offset to
! R10 is different.  However, the only offset we use in the
! minor frame is BSF$A_USER_HAND, so we can simply define
! BSF$MINOR_FRAME as being the same as BSF$MAJOR_FRAME.
!-

MACRO
    BSF$MINOR_FRAME =
 BSF$MAJOR_FRAME %;

!+
! The following are the lengths of the FCD part of the BASIC frame for
! each of the program units.  There are really only three lengths, but
```

```
! we use separate names for many of them to improve the internal
! documentation.
!-

LITERAL
    BSF$K_LENFCDMAJ = 68,                       ! Length of a major frame FCD
    BSF$K_LENFCDMAJ2 = 72,                      ! Length of V2 major frame FCD
    BSF$K_LENFCDDEF = 44,                       ! Length of a DEF function FCD
    BSF$K_LENFCDDFS = 44,                       ! Length of a DEF* function FCD
    BSF$K_LENFCDGSB = 32,                       ! Length of a GOSUB frame FCD
    BSF$K_LENFCDONE = 32,                       ! Length of an ON ERROR frame FCD
    BSF$K_LENFCDIOL = 32;                       ! Length of an IOL frame FCD


!+
! This macro gets the scale factor from the frame of the caller if the caller
! was a BASIC frame.  This macro is used in the BASIC string routines which
! do string to numeric conversions and therefore need to know the scale factor
!-
MACRO
    $BAS$SCALE  =
        BEGIN
        EXTERNAL ROUTINE
            BAS$$SCALE_L_R1 : BAS$SCALE_JSB;
        BUILTIN FP;
        LOCAL FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD);        ! our frame

        FMP = .FP;
        BAS$$SCALE_L_R1 (.FMP [BSF$A_SAVED_FP])
        END %;

!
!       End of file BASFRAME.REQ
!+
```

BASERRMSG REQ

BASFRAME REQ

BASOPN REQ

BPAFSBDEF REQ

BPASTRUCT REQ

BPADABDEF REQ

BPAXRBDEF REQ

BPAERRDEF REQ

BPAFUNDEF REQ

BASINARG REQ

BASBUFSIZ LIS

MATRIX MAR

BASIOERR REQ

BASCB LIS

BPAFQBDEF REQ

BASRTL2 MAP

BASPAR SDL

BASLNK REQ

BPAMSGDEF REQ

BASRTL

BASRTL MAP

BASCANTYP LIS