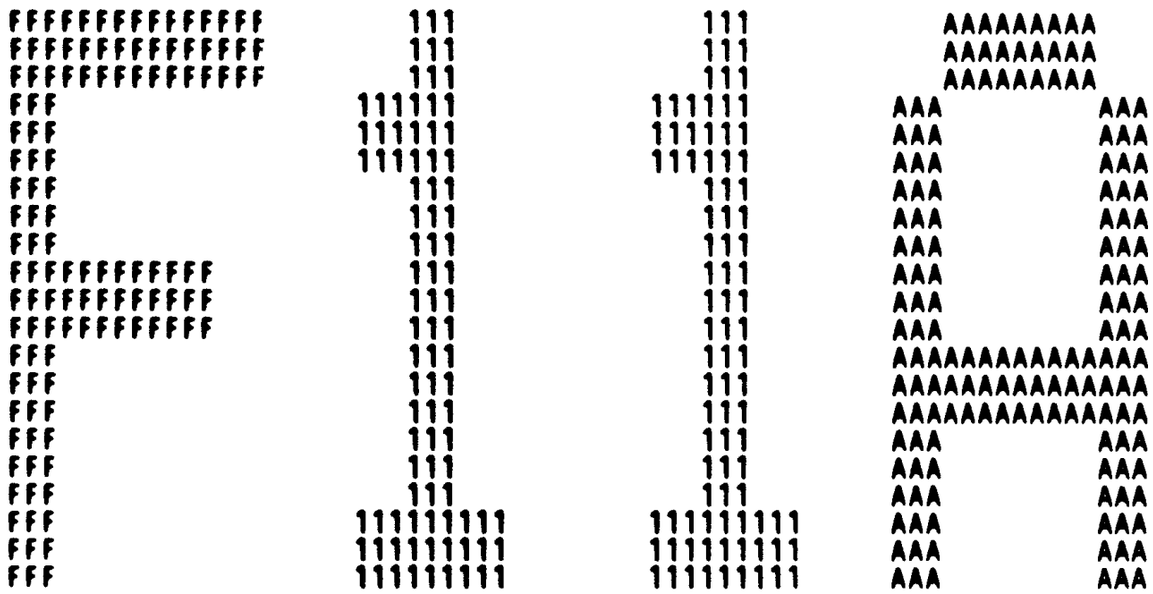
FFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
AAAAAAAAA
AAAAAAAAA
AAAAAAAAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA
AAA AAA

**FILE**ID**FCPDEF

```
FFFFFFFFFF    CCCCCCCC  PPPPPPPP    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
FFFFFFFFFF    CCCCCCCC  PPPPPPPP    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
FF          CC          PP      PP  DD      DD  EE          FF
FF          CC          PP      PP  DD      DD  EE          FF
FF          CC          PP      PP  DD      DD  EE          FF
FF          CC          PP      PP  DD      DD  EE          FF
FFFFFFFF    CC          PPPPPPPP    DD      DD  EEEEEEEE    FFFFFFFF
FFFFFFFF    CC          PPPPPPPP    DD      DD  EEEEEEEE    FFFFFFFF
FF          CC          PP          DD      DD  EE          FF
FF          CC          PP          DD      DD  EE          FF
FF          CC          PP          DD      DD  EE          FF
FF          CC          PP          DD      DD  EE          FF
FF            CCCCCCCC  PP          DDDDDDDD    EEEEEEEEEE  FF
FF            CCCCCCCC  PP          DDDDDDDD    EEEEEEEEEE  FF
```

```
BBBBBBBB      333333      222222
BBBBBBBB      333333      222222
BB      BB  33      33  22      22
BB      BB  33      33  22      22
BB      BB          33          22
BB      BB          33          22
BBBBBBBB          33          22
BBBBBBBB          33          22
BB      BB          33      22
BB      BB          33      22
BB      BB  33      33    22
BB      BB  33      33    22
BBBBBBBB      333333    2222222222
BBBBBBBB      333333    2222222222
```

```
DEFINITION FILE FOR FCP COMPILATION

Version:        'V04-000'
```



```
++
 FACILITY:  F11ACP Structure Level 1

 ABSTRACT:

       These are the data stucture definitions and random macros
       used to compile FCP.

 ENVIRONMENT:

       STARLET operating system, including privileged system calls
       and internal system subroutines.

--


 AUTHOR:  Andrew C. Goldstein, CREATION DATE:  9-Dec-1976  10:53

 MODIFIED BY:

       V03-003 STJ3109         Steven T. Jeffreys,     24-Jun-1983
               Make reference to USER_STATUS general to fix truncation error.

       V03-002 ACG0334         Andrew C. Goldstein,    6-May-1983  14:21
               Fix consistency in declaration of USER_STATUS
```

```
        V03-001 LMP0018         L. Mark Pilant,         31-Mar-1982  11:50
                Add CLF_INCOMPLETE to indicate that the window segment
                is not complete.

        V02-002 LMP0003         L. Mark Pilant,         15-Jan-1982
                Add CLF_REMAP to remap cathedral window

        V02-001 ACG0245         Andrew C. Goldstein,    18-Dec-1981  18:25
                Clean up handling of implicitly spooled files

        V02-000 ACG0167         Andrew C. Goldstein,    7-May-1980  18:50
                Previous revision history moved to F11A.REV
;**
```

```
!
! Declare PSECT usage to minimize page breakage.
!
PSECT
        OWN     = $LOCKEDD1$,
        GLOBAL  = $LOCKEDD1$,
        PLIT    = $CODE$ (EXECUTE);
!
! Declare VAX built in functions.
!
BUILTIN
        TESTBITSC,
        TESTBITCS,
        FFS,
        FFC,
        ROT,
        REMQUE,
        INSQUE,
        CHMU,
        MTPR,
        HALT;
!
! Structure declarations used for system defined structures to
! save typing.
!

STRUCTURE
        BBLOCK [O, P, S, E; N] =
            [N]
            (BBLOCK+O)<P,S,E>,

        BBLOCKVECTOR [I, O, P, S, E; N, BS] =
            [N*BS]
            ((BBLOCKVECTOR+I*BS)+O)<P,S,E>;
!
! Assorted macros used in FCP code
!

MACRO
                                        ! set processor IPL
        SET_IPL (LEVEL) = MTPR (%REF (LEVEL), PR$_IPL)%;
!
! Declare code that must be locked into the working set.
!

MACRO
        LOCK_CODE       =
                PSECT   CODE    = $LOCKEDC1$,
                        PLIT    = $LOCKEDC1$,
                        OWN     = $LOCKEDD1$,
                        GLOBAL  = $LOCKEDD1$;
                        %;
!
! ***** Note: The following two macros violate the Bliss language definition
! ***** in that they make use of the value of SP while building the arg list.
! ***** It is the opinion of the Bliss maintainers that this usage is safe
```

```
! ***** from planned future optimizations.
!
! Macro to call the change mode to kernel system service.
! Macro call format is "KERNEL_CALL (ROUTINE, ARG1, ARG2, ... )".
!
MACRO
        KERNEL_CALL (R) =
            BEGIN
            EXTERNAL ROUTINE SYS$CMKRNL : ADDRESSING_MODE (ABSOLUTE);
            BUILTIN SP;
            SYS$CMKRNL (R, .SP, %LENGTH-1
                        %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI)
            END%;
!
! Macro to call the change mode to exec system service.
! Macro call format is "EXEC_CALL (ROUTINE, ARG1, ARG2, ... )".
!
MACRO
        EXEC_CALL (R) =
            BEGIN
            EXTERNAL ROUTINE SYS$CMEXEC : ADDRESSING_MODE (ABSOLUTE);
            BUILTIN SP;
            SYS$CMEXEC (R, .SP, %LENGTH-1
                        %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI)
            END%;
!
! Macro used to signal fatal errors (internal consistency checks).
!
MACRO
        BUG_CHECK (CODE, TYPE, MESSAGE) =
                BEGIN
                BUILTIN BUGW;
                EXTERNAL LITERAL %NAME('BUG$_',CODE);
                BUGW (%NAME('BUG$_',CODE) OR 4)
                END
                %;
!
! Macro to signal an error status and continue.
!
MACRO
        ERR_STATUS [CODE] =
            BEGIN
            %IF NOT %DECLARED (USER_STATUS)
            %THEN EXTERNAL USER_STATUS : WORD ADDRESSING_MODE (GENERAL)
            %ELSE MAP USER_STATUS : WORD
            %FI;
            IF .USER_STATUS
            THEN USER_STATUS = CODE;
            END%;
!
! Macro to signal an error status and exit.
! Implemented as a change mode to user instruction followed by a RET.
!
MACRO
        ERR_EXIT (CODE) =
            (CHMU (%REF (%IF %NULL (CODE) %THEN 0 %ELSE CODE %FI));
```

```
            RETURN (BUILTIN R0; .R0))
            %;

!
! Macro to generate a string with a descriptor.
!
MACRO
        DESCRIPTOR (STRING) =
                UPLIT (%CHARCOUNT (STRING), UPLIT BYTE (STRING))%;

!
! Macro to return the number of actual parameters supplied to a routine
! call.
!
MACRO
        ACTUALCOUNT =
            BEGIN
            BUILTIN AP;
            .(.AP)<0,8>
            END
            %;

!
! Mode codes for the bad block log file scan routine
!
LITERAL
        REMOVE_BADBLOCK = 0,            ! remove log entry
        ENTER_READERR   = 1,            ! log read error
        ENTER_WRITERR   = 2;            ! log write error

!
! File ID's that are known constants
!
LITERAL
        INDEX_FID       = 1,            ! index file
        BITMAP_FID      = 2,            ! storage map file
        BADBLK_FID      = 3,            ! bad block file
        MFD_FID         = 4,            ! MFD
        CORIMG_FID      = 5,            ! core image file
        VOLSET_FID      = 6,            ! volume set list file
        CONTIN_FID      = 7,            ! continuation file
        BACKUP_FID      = 8,            ! backup journal file
        BADLOG_FID      = 9;            ! bad block log file

!
! Constants used in protection checking
!
LITERAL
        SYSTEM_UIC      = 8,            ! highest UIC group of system UIC's

        READ_ACCESS     = 0,            ! file access modes
        WRITE_ACCESS    = 1,
        DELETE_ACCESS   = 2,
        CREATE_ACCESS   = 3,
        RDATT_ACCESS    = 4,
        WRATT_ACCESS    = 5;

!
! Type codes used to identify blocks being read by READ_BLOCK.
! Note that READ_BLOCK contains a table indexed by these codes.
!
LITERAL
```

```
        HEADER_TYPE     = 0,            ! file header
        BITMAP_TYPE     = 1,            ! storage bitmap
        DIRECTORY_TYPE  = 2,            ! directory block
        INDEX_TYPE      = 3,            ! other index file blocks
        DATA_TYPE       = 4;            ! random data blocks

!
! Type codes used to identify blocks of memory requested from the
! allocator. Note that these codes index into a table in ALLOCATE.
!
LITERAL
        FCB_TYPE        = 0,            ! file control block
        WCB_TYPE        = 1,            ! window block
        VCB_TYPE        = 2;            ! volume control block

!
! Index codes for the subfunctions in the performance measurement data base.
!
LITERAL
        PMS_FIND        = 6,            ! directory searches
        PMS_ENTER       = 7,            ! directory entries
        PMS_ALLOC       = 8,            ! storage map allocation and deallocati n
        PMS_RWATT       = 9;            ! read/write attributes

!
! Random constants.
!
LITERAL
        EFN             = 1,            ! event flag for I/O
        TIMER_EFN       = 2,            ! EFN for timers
        MAILBOX_EFN     = 3,            ! EFN for mailbox I/O
        MIN_WINDOW      = 1,            ! minimum file window size (in pointers)
        MAX_WINDOW      = 80;           ! maximum file window size

!
! Modes to call TRUNCATE routine.
!
LITERAL
        ERASE_POINTERS  = 1,            ! erase retrieval pointers removed
        DEALLOC_BLOCKS  = 1;            ! deallocate the blocks

!
! Normal termination cleanup flags
!
LITERAL
        CLF_FIXFCB      = 1,            ! update FCB from header
        CLF_DOSPOOL     = 2,            ! send file to print queue
        CLF_INVWINDOW   = 4,            ! invalidate all windows
        CLF_SUPERSEDE   = 5,            ! supersede old file
        CLF_SPOOLFILE   = 7,            ! operation is on spool file
        CLF_SYSPRV      = 8,            ! caller is privileged
        CLF_CLEANUP     = 9,            ! cleanup is in progress
        CLF_INCOMPLETE  = 10,           ! window is not complete

!
! Error termination cleanup flags
!
        CLF_DEACCESS    = 16,           ! deaccess file
        CLF_ZCHANNEL    = 17,           ! clean out user's channel
        CLF_TRUNCATE    = 18,           ! undo extend operation
        CLF_CLEANTRUNC  = 19,           ! cleanup after bombed truncate
```

```
        CLF_DELFID      = 20,           ! delete file ID
        CLF_DELFILE     = 21,           ! delete complete file
        CLF_REMOVE      = 22,           ! remove directory entry
        CLF_REENTER     = 23,           ! put directory entry back
        CLF_DELWINDOW   = 26,           ! deallocate window
        CLF_DELEXTFID   = 28,           ! delete extension header
        CLF_REMAP       = 29;           ! do an explicit remap of the file

!
! Cleanup actions that modify the disk, and are to be turned off in case
! of a write error.
!
LITERAL
        CLF_M_WRITEDISK =
                1^CLF_SUPERSEDE         ! supersede old file
                OR 1^CLF_TRUNCATE       ! undo extend operation
                OR 1^CLF_CLEANTRUNC     ! cleanup after bombed truncate
                OR 1^CLF_DELFID         ! delete file ID
                OR 1^CLF_DELFILE        ! delete complete file
                OR 1^CLF_REMOVE         ! remove directory entry
                OR 1^CLF_REENTER        ! put directory entry back
                OR 1^CLF_DELEXTFID;     ! delete extension header
```

CHKSUM
LIS

ACPCNTRL
LIS

CHKPRO
LIS

FCPDEF
B32

DEACCS
LIS

BADSCN
LIS

CLENUP
LIS

CPYNAM
LIS

CHKHDR
LIS

COMMON
LIS

CREHDR
LIS

CREWIN
LIS

ACCESS
LIS

CHKDMO
LIS

DELETE
LIS

CREATE
LIS

CREFCB
LIS