```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
MOTOROLA MICROPROCESSOR & MEMORY TECHNOLOGY GROUP
M68000 Hi-Performance Microprocessor Division
M68060 Software Package
Production Release P1.00 -- October 10, 1994

M68060 Software Package Copyright © 1993, 1994 Motorola Inc.  All rights
reserved.

THE SOFTWARE is provided on an "AS IS" basis and without warranty.
To the maximum extent permitted by applicable law,
MOTOROLA DISCLAIMS ALL WARRANTIES WHETHER EXPRESS OR IMPLIED,
INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE
and any warranty against infringement with regard to the SOFTWARE
(INCLUDING ANY MODIFIED VERSIONS THEREOF) and any accompanying written
materials.

To the maximum extent permitted by applicable law,
IN NO EVENT SHALL MOTOROLA BE LIABLE FOR ANY DAMAGES WHATSOEVER
(INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS,
BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS)
ARISING OF THE USE OR INABILITY TO USE THE SOFTWARE.
Motorola assumes no responsibility for the maintenance and support of the
SOFTWARE.

You are hereby granted a copyright license to use, modify, and distribute the
SOFTWARE
so long as this entire notice is retained without alteration in any modified
and/or
redistributed versions, and that such modified versions are clearly identified
as such.
No licenses are granted by implication, estoppel or otherwise under any patents
or trademarks of Motorola, Inc.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
68060 FLOATING-POINT SOFTWARE PACKAGE (Kernel version)
------------------------------------------------------

The file fpsp.sa contains the 68060 Floating-Point Software
Package. This package is essentially a set of exception handlers
that can be integrated into an operating system.
These exception handlers emulate Unimplemented FP instructions,
instructions using unimplemented data types, and instructions
using unimplemented addressing modes. In addition, this package
includes exception handlers to provide full IEEE-754 compliant
exception handling.

Release file format:
--------------------
The file fpsp.sa is essentially a hexadecimal image of the
release package. This is the ONLY format which will be supported.
The hex image was created by assembling the source code and
then converting the resulting binary output image into an
ASCII text file. The hexadecimal numbers are listed
using the Motorola Assembly Syntax assembler directive "dc.l"
(define constant longword). The file can be converted to other
assembly syntaxes by using any word processor with a global
search and replace function.

To assist in assembling and linking this module with other modules,
the installer should add a symbolic label to the top of the file.
This will allow calling routines to access the entry points
of this package.
```

The source code fpsp.s has also been included but only for documentation purposes.

Release file structure:

```
(top of module)
      -----------------
      |               | - 128 byte-sized section
   (1)  |   Call-Out    | - 4 bytes per entry (user fills these in)
      |               | - example routines in fskeleton.s
      -----------------
      |               | - 8 bytes per entry
   (2)  | Entry Point   | - user does "bra" or "jmp" to this address
      |               |
      -----------------
      |               | - code section
   (3)  ~               ~
      |               |
      -----------------
(bottom of module)
```

The first section of this module is the "Call-out" section. This section is NOT INCLUDED in fpsp.sa (an example "Call-out" section is provided at the end of the file fskeleton.s). The purpose of this section is to allow the FPSP routines to reference external functions that must be provided by the host operating system. This section MUST be exactly 128 bytes in size. There are 32 fields, each 4 bytes in size. Each field corresponds to a function required by the FPSP (these functions and their location are listed in "68060FPSP call-outs" below). Each field entry should contain the address of the corresponding function RELATIVE to the starting address of the "call-out" section. The "Call-out" section must sit adjacent to the fpsp.sa image in memory.

The second section, the "Entry-point" section, is used by external routines to access the functions within the FPSP. Since the fpsp.sa hex file contains no symbol names, this section contains function entry points that are fixed with respect to the top of the package. The currently defined entry-points are listed in section "68060 FPSP entry points" below. A calling routine would simply execute a "bra" or "jmp" that jumped to the selected function entry-point.

For example, if the 68060 hardware took a "Line-F Emulator" exception (vector #11), the operating system should execute something similar to:

```
      bra   _060FPSP_TOP+128+48
```

(_060FPSP_TOP is the starting address of the "Call-out" section; the "Call-out" section is 128 bytes long; and the F-Line FPSP handler entry point is located 48 bytes from the top of the "Entry-point" section.)

The third section is the code section. After entering through an "Entry-point", the entry code jumps to the appropriate emulation code within the code section.

68060FPSP call-outs: (details in fskeleton.s)

```
0x000:      _060_real_bsun
0x004:      _060_real_snan
0x008:      _060_real_operr
0x00c:      _060_real_ovfl
0x010:      _060_real_unfl
0x014:      _060_real_dz
0x018:      _060_real_inex
0x01c:      _060_real_fline
```

```
0x020:      _060_real_fpu_disabled
0x024:      _060_real_trap
0x028:      _060_real_trace
0x02c:      _060_real_access
0x030:      _060_fpsp_done

0x034:      (Motorola reserved)
0x038:      (Motorola reserved)
0x03c:      (Motorola reserved)

0x040:      _060_imem_read
0x044:      _060_dmem_read
0x048:      _060_dmem_write
0x04c:      _060_imem_read_word
0x050:      _060_imem_read_long
0x054:      _060_dmem_read_byte
0x058:      _060_dmem_read_word
0x05c:      _060_dmem_read_long
0x060:      _060_dmem_write_byte
0x064:      _060_dmem_write_word
0x068:      _060_dmem_write_long

0x06c:      (Motorola reserved)
0x070:      (Motorola reserved)
0x074:      (Motorola reserved)
0x078:      (Motorola reserved)
0x07c:      (Motorola reserved)

68060FPSP entry points:
-----------------------
0x000:      _060_fpsp_snan
0x008:      _060_fpsp_operr
0x010:      _060_fpsp_ovfl
0x018:      _060_fpsp_unfl
0x020:      _060_fpsp_dz
0x028:      _060_fpsp_inex
0x030:      _060_fpsp_fline
0x038:      _060_fpsp_unsupp
0x040:      _060_fpsp_effadd



Miscellaneous:
--------------

_060_fpsp_snan:
----------------
- documented in 3.5 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_snan --|
                                        |
            always exits through _060_real_snan <----

_060_fpsp_operr:
----------------
- documented in 3.5 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_operr --|
                                         |
           always exits through _060_real_operr <-----

_060_fpsp_dz:
----------------
- documented in 3.7 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_dz   --|
                                             |
            always exits through _060_real_dz   <----

_060_fpsp_inex:
----------------
- documented in 3.6 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_inex --|
                                             |
            always exits through _060_real_inex <----


_060_fpsp_ovfl:
----------------
- documented in 3.4 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_ovfl --|
                                             |
            may exit through _060_real_inex     <---|
                                          or |
            may exit through _060_real_ovfl     <---|
                                          or |
            may exit through _060_fpsp_done     <---|

_060_fpsp_unfl:
----------------
- documented in 3.4 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_unfl --|
                                             |
            may exit through _060_real_inex     <---|
                                          or |
            may exit through _060_real_unfl     <---|
                                          or |
            may exit through _060_fpsp_done     <---|


_060_fpsp_fline:
-----------------
- not fully documented in 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_fline --|
                                              |
            --------------------------------------------
            |               |                     |
            v               v                     v
   (unimplemented   (fpu disabled)       (possible F-line illegal)
     stack frame)           |                     v
            |               v                  special case "fmovecr"?
            |         exit through                    |
            |     _060_real_fpu_disabled        -------------
            |                                   |           |
            |               ^                   v           v
            |               |                 (yes)       (no)
            |               |                   v           v
            |               |             fpu disabled?   exit through
            |               |                   |         _060_real_fline
            v               |             -------------
            |               |             |           |
            |               |             v           v
            |               |-----------(yes)        (no)
            |                                         |
            |----<------------------------------------|
            |
            |
            |----> may exit through _060_real_trace
            |
            |----> may exit through _060_real_trap
            |
            |----> may exit through _060_real_bsun
            |
            |----> may exit through _060_fpsp_done


_060_fpsp_unsupp:
------------------
- documented in 3.1 of 060SP spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_unsupp --|
                                          |
                                          |
            may exit through _060_real_snan      <----|
                                     or   |
            may exit through _060_real_operr     <----|
                                     or   |
            may exit through _060_real_ovfl      <----|
                                     or   |
            may exit through _060_real_unfl      <----|
                                     or   |
            may exit through _060_real_inex      <----|
                                     or   |
            may exit through _060_real_trace     <----|
                                     or   |
            may exit through _060_fpsp_done      <----|


_060_fpsp_effadd:
------------------
- documented in 3.3 of 060 spec.
- Basic flow:
      exception taken ---> enter _060_fpsp_effadd --|
                                          |
                                          |
            may exit through _060_real_trace     <----|
                                     or   |
     may exit through _060_real_fpu_disabled     <----|
                                     or   |
            may exit through _060_fpsp_done      <----|
```