Innovative Systems

Floating Point Engine (FPE) Owner's Manual

1989-91

Copyright

Innovative Systems
P.O. Box 444
Severn, MD 21061-0444
(301)987-8688/768-4599

Limited Warranty on Media and Replacement

If you discover physical defects in the manuals distributed with an Innovative Systems product or in the media on which a software product is distributed, Innovative Systems will replace the media or manuals at no charge to you, provided you return the item to be replaced with proof of purchase to Innovative Systems or an authorized Innovative Systems dealer during the 90-day period after you purchased the software.

ALL IMPLIED WARRANTIES ON THE MEDIA OR MANUALS, INCLUDING IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF THE ORIGINAL RETAIL PURCHASE OF THE PRODUCT.

Even though Innovative Systems has tested the software and reviewed the documentation, **INNOVATIVE SYSTEMS MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO SOFTWARE, ITS QUALITY, PERFORMANCE, MERCHANTIBILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS SOFTWARE IS SOLD "AS IS," AND YOU THE PURCHASER ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE.**

IN NO EVENT WILL INNOVATIVE SYSTEMS BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE OR ITS DOCUMENTATION, even if advised of the possibility of such damages. In particular, Innovative Systems shall have no liability for any programs or data stored or used with Innovative Systems products, including the costs of recovering such programs or data.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Innovative Systems dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Warning

Innovative Systems provides a Floating Point Engine card that is for installation in your personal computer. Thus, the FPE is classified as a subassembly by the FCC. See instructions if interference to radio or television reception is suspected.

Information to Users

This floating point card generates and uses radio frequency energy and if not installed and used properly – that is, in strict accordance with the manufacturer's instructions – may cause interference to radio and television receptions.

Instructions

If this card does cause interference to radio or television reception – which can be determined by turning the equipment on and off and noting the effect of the power surge on the radio or television – you are encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna.
- Move the computer away from the receive.
- Plug the computer into a different outlet so that the computer and receiver are on different branch circuits.

If necessary you should consult with Innovative Systems or an experienced radio/television technician for additional suggestions. You may find the following booklet prepared by the FCC helpful: "How to Identify and Resolve Radio-TV Interference Problems." This booklet is available from the U.S. Government Printing Office, Washington, D.C., 20403, Stock No. 004-000-00345.4.

LIMITED WARRANTY

Innovative Systems warrants all of its hardware products, including spare parts sold by Innovative Systems, to be free from defects in material and workmanship for a period of five years from the date of delivery.

This warranty is made to original purchasers only, and only original purchasers make make any claim under the warranty. No other party shall have any rights under this warranty. The sole remedy for any breach of this warranty shall be the repair or replacement of the defective product, as described herein.

Innovative Systems disclaims all other representations and warranties, included but not limited to, any implied warranty of merchantibility or fitness for a particular purpose. Innovative Systems shall not be liable for any special, indirect, incidental or consequential damages, lost profits, costs or expenses, except as set forth in this policy, which may be modified or amended only by written contract.

In-Warranty Repair

Innovative Systems will repair at its factory or repair center, any product that within the warranty period is returned to Innovative Systems and found to be defective in proper usage.

Innovative Systems will honor the warranty if notification of product failure is provided within the five year warranty period. The original customer must return the defective product to Innovative Systems. One-way transportation charges are at the customer's expense. Innovative Systems will return the repaired or replaced product at the expense of Innovative Systems.

Innovative Systems reserves the right to reject any warranty claim on any products that have been the subject of abuse, misuse, unauthorized repair, alteration, accident, improper return handling or causes external to the

product but not limited to: improper power application, improper environmental exposure or other improper use of the product.

Innovative Systems includes in its Limited Warranty policy, provisions for updating in accordance with any field change order which Innovative Systems determines is mandatory for reasons of product safety. All other field changes, revisions or updates not deemed mandatory by Innovative Systems may be implemented at the discretion of Innovative Systems or as required by contract.

Out-of-Warranty Repair

Innovative Systems will provide repair or replacement services for all products manufactured by or for Innovative Systems and sold by Innovative Systems for a reasonable active product support period extending beyond last date of standard manufacture and sale. This period will normally be for a period of two years from Innovative Systems standard product list, but such period may be decreased at Innovative Systems' sole option.

Out-of-warranty products and customer-related damage of in-warranty products will be repaired or replaced in accordance with Innovative Systems' then-current active product repair price schedule. The customer is obligated for freight and handling charges both ways.

Below are the prices for Out-of-Warranty products manufactured or sold by Innovative Systems. The prices are effective 1 June 1988 and are subject to change without notice.

FPE
$25.00 plus 10.00
Shipping, Handling and Processing Charges.
Our rate does not include parts.

Repair Warranty

Innovative Systems warrants any product repaired in its factory or repair center to be free from defects in material and workmanship for a period of ninety (90) days from the date of return delivery or the end of the original warranty period, whichever is greater.

Warranty Registration

Please take a moment to fill out the warranty registration form within ten days and mail it to the following address:

Innovative Systems
P.O. Box 444
Severn, Maryland 21144-0444

Attn: Customer Service

TABLE OF CONTENTS

1. Introducing the Floating Point Engine™

The Innovative Systems™ (iS™) Floating Point Engine™ (FPE™) provides the most efficient floating point math capability for all members of the Apple II™ family. Based on the Motorola MC68881 floating point processor, the FPE brings a new dimension in computing power to the Apple II. The MC68881 is the same floating point processor used with the Motorola 68000 microprocessor. Although you may need not be concerned with specific capabilities of the FPE, software and programmers have access to:

* Eight general purpose, 80-bit floating-point data registers.

* Forty-six instructions, including 35 arithmetic operations.

* Full ANSI-IEEE 754-1985 floating point standard.

* Enhanced functions, including a complete set of trigonometric and transcendental functions.

* Seven data formats: byte, word, and long word integers; single, double, and extended precision real numbers; and packed binary coded decimal string real numbers.

* Twenty-two constants including pi, e, and powers of 10.

* Concurrent instruction execution with the Apple II.

The FPE may be called in several ways. If the system software automatically loads all tools from your disk, then the FPE is directly callable from the Apple IIGS™ Standard Apple Numerics Environment (SANE™) toolset. You need only boot the FPETOOLS disk to install the FPE software called FPETOOL.INIT onto your system disk in the /SYSTEM/SYSTEM.SETUP directory. All calls intended for SANE automatically call the FPE once the system is rebooted from a complete shutdown. Thus, the FPE is transparent to you, except in terms of speed improvement. This technique works only with the GS series.

For those users who have Apple II, II+, or //e computers, the FPETOOLS disk contains a version of 8-bit SANE which addresses the FPE. This version of SANE replaces all calls except those calls to the Scanner and Formatter operations (FPSTR2DEC, FCSTR2DEC, and FDEC2STR). For further information, refer to Section 4.2.

For higher performance the FPE may be directly addressed through software by writing directly to the command or reading directly from the status registers, as appropriate, in normal slot space ($C0nx, where n=8 plus the slot number). This technique works equally well with any Apple II, DOS 3.2, DOS 3.3, ProDOS 8™, ProDOS 16, and GS/OS™ without the overhead of using a toolset. Please refer to Chapters 3 and 5.

The FPE is also compatible with all versions of AppleWorks Classic. You need only boot the FPETOOLS installation disk to install FPE software software patches to AppleWorks. These patches will provide a significant improvement in the calculation and recalculation time required by the spreadsheet and some database operations.

This manual describes how to communicate with the Innovative Systems FPE. It does not explain the inner workings of the Motorola MC68881 floating point coprocessor. Refer to the Motorola "MC68881/MC68882 Floating Point Coprocessor User's Manual", and related application notes, for details on how to use the MC68881. These items are available from Motorola or, for a nominal charge, from Innovative Systems.

This manual is written to address different levels of users. If you do not plan to write your own code, you need to read Chapters 1 and 2 only. If you intend to address the FPE using your own code, you will also need to read and to understand Chapters 3 through 10.

2. Installing the FPE.

Installation consists of three parts: hardware, software and slot enabling.

Hardware installation consists of plugging the FPE into an expansion slot in the Apple II. This means that you may use any slot numbered between 1 and 7. Don't try to use the memory expansion slot in the IIGS-it is not a peripheral slot. The slot you choose will be dictated by the slots you have available.

Note that there are only two ways you can damage the board during installation - static electricity and putting the board in backwards. If you carefully observe the following instructions, neither will be a problem:

1. **Ensure all power to your computer is off by removing the power cord from the wall outlet.**

2. Carefully remove the case cover from your computer as described in the owner's manual supplied by Apple.

3. Face the computer as you normally would if using it (keyboard toward you. Refer to Figure 2-1.).

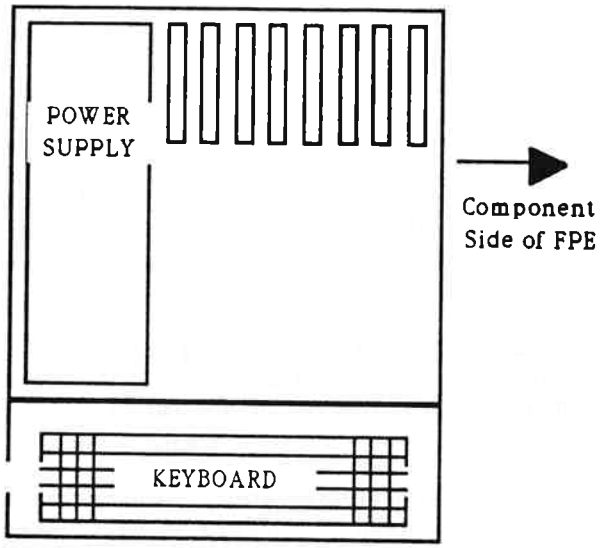


FIGURE 2-1 FPE Installation

4. Ground yourself by touching the top of the metal cover of the power supply on the left hand side of the computer.

5. Remove the FPE from the box and the anti-static plastic wrapping.

6 Plug the FPE into the slot of your choice, ensuring that the component side of the board (the side with the lettering) faces to your right, away from the power supply.

7. Replace the case cover, plug the computer power cord into the power outlet, apply power, and boot your computer as normal.

8. The computer should boot normally.

9. The computer is now ready for software installation.

2.2. Software

Software installation requires booting the FPETOOLS distribution disk. The software on this this disk will move certain files from the iS installation disk to your system disk.

1. Enable the slot if you have a IIGS and the slot requires enabling to use "YOUR CARD" (see Slot Enabling). Then power down the computer.

2. Boot the FPETOOLS installation disk. The disk will automatically locate the FPE and report the slot number to you.

3. Select option 3 from the Installation Menu. This test will verify that the FPE functions correctly.

NOTE:

1. If the installation software reports an error or your system hangs, verify that you have installed the FPE correctly (power down your system first, then reboot) and verify that you have enabled the slot (set it to "Your Card"), if you have an Apple IIGS. If the rerunning the test returns an error or your system hangs, please contact Innovative Systems for technical support.

2. If the test does not return (the system hangs) and you have an Applied Engineering TransWarp GS™, you will need to contact AE to obtain a modification to the TWGS. Contact AE customer support for more information.

4. If you have a IIGS, select option 1 too install the FPE Toolset on your system disk(s). Note that if your system disk is named "/hard1", for example, the disk name you should enter when requested is "/hard1". No directory information is required. After successful installation, all SANE calls (TOOL010 or $xxxA) will then automatically access the FPE without any need to recompile, reconfigure, or replace any of your existing commercial or user-written software that uses the SANE Toolset.

5. If you have AppleWorks Classic, select option 2 to install a patch which provides the capability for AppleWorks to use the FPE when doing math. Because AppleWorks can be in a subdirectory, please provide the volume name and the subdirectory in response to the prompt from the initialization program. For example, if AppleWorks is located in directory "/AppleWorks" on volume "/hard1", please enter "/hard1/AppleWorks" when prompted. Also, if your Startup disk and your Program disk are the same, enter the same information after both prompts.

6. Select option 0 to exit the installation program.

Innovative Systems has provided a FPE toolset initialization file, coded specifically for each slot. These slot dependent files provide a small speed improvement over files which automatically locate the FPE slot, because the code uses direct addressing of the FPE slots rather than using indirect indexed addressing. Thus, if the FPE is in slot 2, you must have the file "FPETOOL.INIT.S2" in your "/SYSTEM/SYSTEM.SETUP" directory on your startup disk.

NOTE: USE OF ANY FPETOOL.INIT FILE NOT CORRESPONDING TO THE SLOT NUMBER CONTAINING THE FPE WILL CRASH YOUR SYSTEM.

Optimized code for accessing the FPE from a higher level language will be included in the particular software package you purchase (such as ORCA/C) and requires no installation on your part.

The iS FPETOOLS installation disk also includes sets of macros (M8.FPE and M16.FPE), definitions (E8.FPE and E16.FPE), and some examples for various development packages (APW, ORCA/M, MERLIN 8/16, MERLIN 16+, LISA816) for those who wish to write their own code. Separate macro libraries are provided for the 6502/65C02 and 65816 microprocessors. The user should use the macro library appropriate for his computer. These files may reside anywhere on the user's disk. The macros are included in the folders "FPE.IIGS" and "FPE.6502" on the installation disk.

2.3. Slot Enabling on the Apple IIGS

You may have to enable the slot in which the FPE is installed. Follow the instructions in your user's manual to use the control panel to select "Your Card" for the appropriate slot. If you install the FPE in slot 3 (recommended), you do not need to enable the slot as it is always properly enabled.

2.4. Slot Enabling on the Apple II, II+, and //e

No enabling is required for Apple II, II+, or //e computers because the slot I/O is normally active.

3. Access to the FPE

How does the System know which slots contains the FPE?

1. If you have an Apple IIGS and you have loaded the FPETOOL.INIT file corresponding to the slot containing the FPE, all calls to SANE will automatically go to the FPE.

2. If you write your own code to directly access the FPE, you must use the correct address for the slot locations: that is,

   $c080 + 16*slot_number (e.g., $c090 for slot 1).

   Refer to Chapter 9 for information on how to determine the FPE slot number without hard coding the slot number into your code.

Direct access means that software writes information directly to or reads data directly from the FPE coprocessor interface registers. These interface registers reside in the 16 locations reserved for the slot in which the card resides. These 16 locations are designated as Read-only, Write-only, or Read/Write, depending upon their purpose. In using direct access, the software does not need to "pass through" unnecessary general purpose code.

Direct access is the most efficient method of communicating with the FPE. It eliminates overhead; this is not to say it is always the best method of interfacing, however. Direct access programming requires a strong understanding of programming. Chapters 5 and 6 contain additional information necessary to do direct accessing of the FPE.

The Motorola MC68881 communicates with the host processor (6502, 65C02, or 65816) by way of Coprocessor Interface Registers (CIR). These registers are used for control of, transferring operands to, and returning status from the MC68881. The Apple II technical manuals and the Motorola "MC68881/68882 Floating-Point Coprocessor User's Manual" contain valuable information on accessing the registers and details which explain the uses for the CIRs. The iS FPE allows access to all the CIRs that are required to implement all MC68881 instructions. The only CIRs not accessible are those intended for use with the 68020/68030 microprocessors, and which do not impact performance with the 6502/65816. The registers implemented in the FPE and their base addresses are given in Table 3-1.

Table 3-1: Coprocessor Interface Register (CIR) Memory Map

Base Address Register	Location	Width	Type
Response	$C0k0	16	R
Control	$C0k2	16	W
Save	$C0k4	16	R
Restore	$C0k6	16	R/W
Command	$C0k8	16	W
Condition	$C0kA	16	W
Operand	$C0kC	32	R/W

1. All transfers are byte swapped from normal 6502/65816 storage; that is, the MSB of the data is contained in the lowest memory address.
2. k is the number of the slot containing the FPE + 8.
3. Word transfers (16 bits) to the Operand register use addresses $C0kC and $C0kD. Multiple word transfers (32, 64, 80, and 96 bits) use all four locations ($C0kC-$C0kF). Note that for 80-bit transfers, the first data transfer requires that $C0kE and $C0kF receive $0 values and that bits 65 to 80 are transferred to $C0kC and $C0kD.
4. All locations are located in the I/O page ($00 or $E1) of 65816 RAM space.
5. All locations are in page $C0 of the 6502 RAM space.

Remember that the register addresses are base addresses; so the address for a specific slot is specified by replacing the k in the base address with 8 + slot number. For example, if the FPE is in slot 3, the Response register starts at $C0b0. Additional information on peripheral card addressing is available in Chapter 6, "Programming for Peripheral Cards", pages 129-131 and 136-137 of the "Apple IIe Technical Reference Manual."

4. Interfacing to SANE

The Standard Apple Numerics Environment™ (SANE) defines a series of calls which provide numeric operations in accordance with IEEE Standard 754 Binary Floating-Point Arithmetic. SANE also provides several utility functions which include conversions of data from an ASCII representation to binary floating point and back again. This environment provides very accurate numerics. Unfortunately, SANE operations can be very slow. The iS FPE provides the numeric operations, but at a much faster rate.

Because SANE is standard with the Apple II computer, Innovative Systems provides numerics software package which replaces most of the routines in the SANE toolset. The software uses the same calling sequences, processes the commands in 80-bit precision, and generally provides the same results as those described in the "Apple Numerics Manual", available from your Apple dealer. One difference is that the transcendentals returned are slightly less accurate (76 bits or more of accuracy versus 80 bits from SANE); however, this change in accuracy should not adversely affect the performance of your software (see "Apple Numerics Manual, Second Edition", Chapters 28, and Chapter 10 of this manual for the details). Another difference is that the FPE does not process COMP type variables; however, COMP calls will work with the FPE toolset (except at the speed of the Apple II since the calls use the standard SANE code). Because the FPE toolset is a hybrid of calls to the FPE and to the standard SANE toolset code, use of the FPE toolset is automatic and transparent to most existing software.

4.1 Apple IIGS

To use the iS numerics software on an Apple IIGS, your must have copied the FPE.INIT from the iS source disk to the /SYSTEM/SYSTEM.SETUP subdirectory on the system disk. This is normally done by the FPETOOLS distribution disk.

4.2 Apple II, II+, and //e

The replacement for the SANE interface in the Apple II,II+, or //e is customized (to a specific absolute memory address) and is included on the FPETOOLS distribution disk in the "/FPETOOLS/FPE.6502/TOOLSET" directory. This toolset uses the following calls:

jsr	$2100	to call the fp6502 routines
jsr	$2104	to call the ELEMS6502 routines.

This toolset loads into locations beginning at $(00)2100 and has a length of less than $1000 bytes. The toolset has a filetype of BIN.

For more information, please refer to the "Apple Numerics Manual" available from Addison-Wesley Publishing Company, Inc.

4.3 AppleWorks™ Classic

The replacement for the AppleWorks Classic calls to the 8-bit SANE software is included on the FPETOOLS distribution disk.

4.2 AppleWorks™ GS

Support for AppleWorks GS is automatically provided as this package uses the GS/OS and ProDOS 16 SANE tool set calls.

The Innovative Systems FPE fully supports Motorola's MC68881 coprocessor dialog. The dialog consists of a rigidly structured combination of commands and response primitives. The commands tell the MC68881 what to do, and the primitives indicate actions that are required, including: transfer data, wait for synchronization, wait for completion of operation, and handle error conditions. Failure to follow the coprocessor protocol can result in destruction of your code during program execution.

The FPE allows three types of operations: Memory-to-Register (MEMREG), Register-to-Memory (REGMEM), and Register-to-Register (REGREG). MEMREG and REGMEM operations may be done at any precision. REGREG operations are always done in extended precision.

5.1 MEMREG and REGMEM Operations

MEMREG and REGMEM operations move data from Apple memory to a MC68881 floating point, control, or status register, and from a MC68881 floating point, control, or status register to Apple memory (refer to the flow charts in Figures 5-1 and 5-2). These operations are often called move-in or move-out operations, respectively. They require that the software

1. Write a command word (16 bits) to the Command register ($C0k8)

2. Check the word in the Response register ($C0k0) for a Null Come-Again (CA) (any value other than ($8900)

3. Transfer the operand byte(s) to or from the Operand register ($C0kC)

4. Check the word in the Response register ($C0k0) for a Null Release (i.e., the most significant bit (CA bit) is equal to 0)

The $8000 and $8900 signify that the values are written the way the MC68881 expects to write them; however, the 6502/65816 must read and write all data in byte reversed order ($0089 in this case). The reason for the byte reversal is that the 6502 and the 65816 write the low byte of the accumulator to the low byte of memory or to a peripheral slot. This is opposite to the requirements of the MC68xxx series. Hence, the 68881 expects or reports the most significant byte (MSB) as the low address byte. You must transpose the byte order of all data (including 80 bit data) to satisfy the MC68881. Remember this because it applies to every command or operand transfer to, and operand and response transfer from, the FPE.

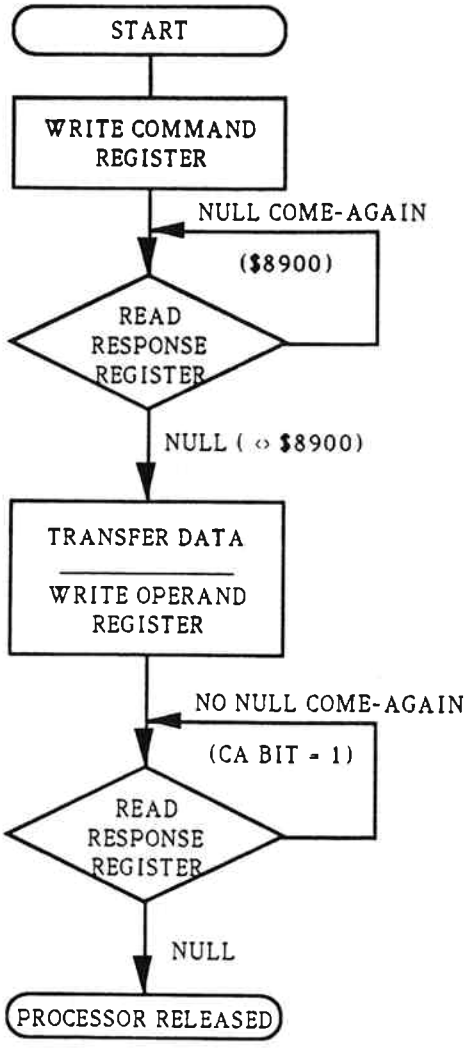


FIGURE 5-1. MOVE-IN SEQUENCE (MEMREG)

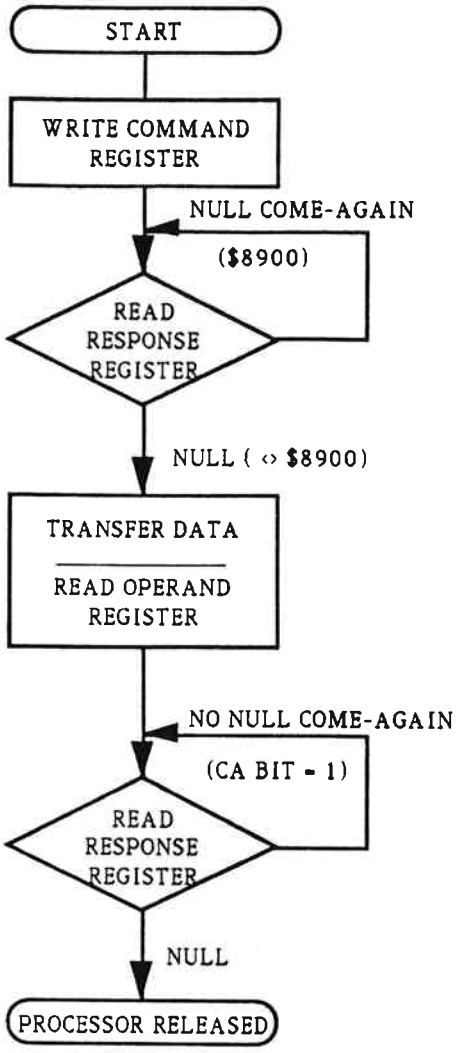


FIGURE 5-2. MOVE-OUT SEQUENCE (REGMEM)

You might even be wondering why we check for a value of $8900. The answer is adaption. If the MC68881 was being used with an MC68020 microprocessor, the value read from the Response register would indicate the number of bytes to be transferred. In FPE applications, the MC68881 does the same, but the 6502/65816 cannot easily make sense of this value. So to improve processing time, Innovative Systems noted that $8900 is the only response primitive that requires the 6502/65816 to wait before transferring data. Any other value from the Response register of the iS FPE implementation indicates that the 6502/65816 may transfer an operand. **Warning**: don't try to test for a non-$8900 value as this will confuse the MC68881 and destroy any data in the FPE.

5.2 REGREG Operations

REGREG operations are used for operations that do not require operand data from memory to register transfers (refer to Figure 5-3). Examples include adding two registers (both registers having a data value), taking the sine of a value in a register, or even transferring a constant value from the ROM internal to the MC68881 to a register. The sequence of operations is:

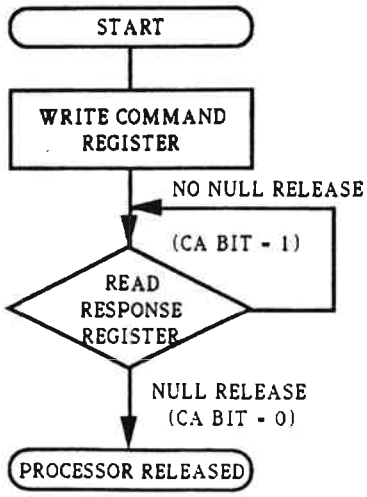


FIGURE 5-3. REGISTER/REGISTER SEQUENCE (REGREG)

1. Write the command to the Command register ($C0k8)

2. Check the Response register for a Null Release ($C0k0)

Since there are no external operands, a REGREG operation does not require that the software test for a Null Response in the Response register as the MEMREG and REGMEM operations do. Once it has written the command to the Command register (with correct byte order), the software need only test the Response register
for the Null Release.

NOTE: Don't try to test for a non-$8900 value as this will confuse the MC68881 and destroy any data in the FPE.

5.3 Checking Status

Example code segments for checking the status from the FPE are as follows:

65816 Version

```
                ldy     #response
loop1           lda     [<mc68881],y    assumes the location containing
;                                       the base address of the FPE is
;                                       in the direct page
                cmp     #$0089          check for Null Come-Again
                beq     loop1


                ldy     #response
loop2           lda     [<mc68881],y    check for Null Release
                and     #$0080
                bne     loop2
```

6502 Version

```
loop1           ldy     #response
                lda     (mc68881),y      check for Null Come-Again
;                                        (location containing
;                                        the base address of the FPE is
;                                        somewhere in memory,
;                                        location designated by
;                                        mc68881)
                tax
                iny
                lda     (mc68881),y
                bne     continue
                cpx     #$89
                beq     loop1
continue
```

```
loop2   ldy     #response
        lda     (mc68881),y     check for Null Release
        iny
        lda     (mc68881),y     always must read upper byte
        asl     a
        bcs     loop2
```

6. Construction of an MC68881/MC68882 Command

Each command written directly to the floating point coprocessor Command register requires 16 bits of information. The format for the command (as seen by the MC68881) is:

MSB															LSB
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	R/M	0	S	S	S	D	D	D	0	0	C	C	C	C	C

where [R/M] Field – Specifies the source operand address mode.
0 – The operation is register to register.
1 – The operation is memory to register or register to memory.
[SSS] (Source Specifier Field) – Specifies the source register or data format.
If R/M = 0, specifies the source floating point data register, FPm.
If R/M = 1, specifies the source data format:
000 L Long Word Integer (32-bits)
001 S Single Precision Real (32-bits)
010 X Extended Precision Real (96-bits)1
011 P Packed Decimal Real (96-bits)2
100 W Word Integer (16-bits)
101 D Double Precision Real (64-bits)
110 B Byte Integer (8-bits)
[DDD] (Destination Register) – Specifies the destination floating point register, FPn.
[CCCCC] (Execution Command) – Specifies the operation to perform.

NOTE
1. Only 80 bits contain valid data, but 96 bits must be transferred.
2. Only 84 bits contain valid data, but 96 bits must be transferred.
3. See "MC68881/MC68882 Floating-Point Coprocessor User's Manual", pages 3-1, 3-2, and 3-7 for format information)
4. All operations which input data to the FPE transfer information from the source (SSS or memory) to the destination register (DDD). This means that the source value is moved (e.g., added) to the destination register.
5. All register-to-register operations move data from the source register to the destination register (e.g., the source register is added to the contents of the destination register).

The files E16.FPE and E8.FPE contain definitions for the R/M and Source Specifier fields, the Destination Register field, and the Execution Command field. To define a command to add an extended real number to register 1 do the following:

1. Get the Memory-to-Register Extended Precision value from the definitions table (Table 6.1-1)

2. Get the value for Floating Point Register 1 from Table 6.1-2 (%001). Put this value into the Destination register field (DDD, bits 7-9). The command word should now be $4880.

3. Put the value for the command (FADD in Table 6.1-3) into bits 0-4. From the definition file, FADD equals $22. The command word should now be $48A2. Remember that the word is in reverse order as seen from the Apple computer, so reverse the data bytes. The value for the command is, therefore, $A248.

Similarly, a register 1 (SSS value) to register 2 (DDD value) add would have a final command value of "%0010001000000101" or $2205 in Apple memory.

NOTE
1. To make this process easier, iS has supplied macro files which will generate the most used commands for you.
2. The 16-bit binary values for commands are given in non-Apple memory order in the examples associated with Tables 6.1-3 and 6.1-4.

Table 6.1-1 MC68881 Command Primitives

Register-to-Memory Movement	
Single Precision	$6400
Long Integer	$6000
Word Integer	$7000
Byte Integer	$7800
Double Precision	$7400
Extended Precision	$6800
Packed BCD	$6c00*
Memory-to-Register Movement	
Single Precision	$4400
Long Integer	$4000
Word Integer	$5000
Byte Integer	$5800
Double Precision	$5400
Extended Precision	$4800
Packed BCD	$4c00
Register-to-Register Movement	
Extended Precision (only)	$0000
Constant in ROM-to-Register Movement (see Table 6.1-4)	
Extended Precision (only)	$5c00
Memory-to-Control, Status or Instruction Register	
Long Integer (only)	$0000
Control, Status or Instruction Register-to-Memory	
Long Integer (only)	$2000

* The retrieval of a packed BCD value form the FPE requires a formatting value (k-factor). The k-factor format is as follows (encoded twos complement integer (3-bits in locations 3-5)):

-64 to 0 - indicates the of significant digits to the right of the decimal point (FORTRAN F format)
+1 to +17 - indicates the number of significant digits in the mantissa (FORTRAN E format)
+17 to +63 - treated as +17

Table 6.1-2 Register Values

Floating Point Register 0	%000
Floating Point Register 1	%001
Floating Point Register 2	%010
Floating Point Register 3	%011
Floating Point Register 4	%100
Floating Point Register 5	%101
Floating Point Register 6	%110
Floating Point Register 7	%111
Control Register	$9000
Status Register	$8800
Instruction Address	$8400

Table 6.1.-3 Operations Values

FMOVE	$00	Move
FINT	$01	Integer Part
FSINH	$02	Hyperbolic sine
FSQRT	$04	Square Root
FLOGNP1	$06	LOGe(1+X)
FETOXM1	$08	((e**X)-1)
FTANH	$09	Hyperbolic tangent
FATAN	$0a	Arctangent
FASIN	$0c	Arcsine
FATANH	$0d	Hyperbolic arctangent
FSIN	$0e	Sine
FTAN	$0f	Tangent
FETOX	$10	e**X
FTWOTOX	$11	2**X
FTENTOX	$12	10**X
FLOGN	$14	Natural log
FLOG10	$15	Log base 10
FLOG2	$16	Binary log
FABS	$18	Absolute Value
FCOSH	$19	Hyperbolic cosine
FNEG	$1a	Negate
FACOS	$1c	Arccosine
FCOS	$1d	Cosine
FGETEXP	$1e	Get exponent
FGETMAN	$1f	Get mantissa
FDIV	$20	Divide
FMOD	$21	Modulo Remainder
FADD	$22	Add
FMUL	$23	Multiply
FSGLDIV	$24	Single precision divide
FREM	$25	IEEE Remainder
FSCALE	$26	Scale exponent
FSGLMUL	$27	Single precision multiply
FSUB	$28	Subtract
FCMP	$38	Compare SSS with DDD
FTST	$3a	Test
FSINCOS	$30	Simultaneous sine and cosine*

*FSINCOS requires three registers, one source and two destination, and is a register-to-register operation only. The form for this command is:

%00SSSDDD0000ddd + operation

where: ddd = destination 2 register (cosine value)
DDD = destination 1 register (sine value)
SSS = source register

Table 6.1-4 Constant in ROM-to-Register Values

$00	PI
$0b	LOG10(2)
$0c	e
$0d	LOG2(e)
$0e	LOG10(e)
$0f	0.0
$30	LOGn(2)
$32	10**0
$33	10**1
$34	10**2
$35	10**4
$36	10**8
$37	10**16
$38	10**32
$39	10**64
$3a	10**128
$3b	10**256
$3c	10**512
$3d	10**1024
$3e	10**2048
$3f	10**$4096

uses form: %010111DDD00vvvvv

where: DDD = destination
vvvvv = ROM value.

7. Macro Usage

The iS FPE comes with macro library files. These files are compatible with the APW, ORCA/M, LISA816, and MERLIN assemblers. M16.FPE, in conjunction with the E16.EQU file, (for LISA816 use only M16.68881) contains macros for use with the 65816 microprocessor in the Apple IIGS. M8.FPE contains the macros for the 6502-based Apple computers. These macros are assembler specific and are contained in folders labeled for the appropriate assembler.

The macros define the command for each operation desired. You just need to supply the operation wanted, the address of the correctly formatted data, and the register(s) to use. These macros will load or retrieve the results of the operation. The general format of the macros is as follows:

APW/ORCA/LISA816 Assembly

Memory-to-Register:

```
MEMREGv     OPERATION_CODE,DESTINATION_FP_REGISTER,DATA_ADDRESS
```

where v = precision of operation (X, D, S, L, W)

Register-to-Memory:

```
REGMEMv     OPERATION_CODE,SOURCE_FP_REGISTER,DATA_ADDRESS
```

where v = precision of operation (X, D, S, L, W)

Register-to-Register:

```
REGREG    OPERATION_CODE,SOURCE_FP_REGISTER,DESTINATION_FP_REGISTER
```

MERLIN Assembly

Memory-to-Register:

```
MEMREG   PRECISION;OPERATION_CODE;DESTINATION_FP_REGISTER;DATA_ADDRESS
```

where PRECISION = X, D, S, L, W

Register-to-Memory:

```
REGMEM   PRECISION;OPERATION_CODE;SOURCE_FP_REGISTER;DATA_ADDRESS
```

where PRECISION = X, D, S, L, W

Register-to-Register:

```
REGREG    OPERATION_CODE,SOURCE_FP_REGISTER,DESTINATION_FP_REGISTER
```

The source code below is an example of macro usage and shows the form for code which uses the FPE.

```
**************************************************
*
*       SAMPLE TASK FOR ADDING TWO EXTENDED
*       PRECISION NUMBERS, SHOWING THE USE
*       OF MACROS.
*
**************************************************
               MLOAD       2/AINCLUDE/M16.UTILITY
               MLOAD       M16.FPE
TEST           START
               COPY        E16.FPE
MC68881        EQU   $00             DIRECT PAGE LOCATION OF FPE
;                                          BASE REGISTER
               CLC
               PHK
               PLB
               STZ         $00               ZERO DIRECT PAGE DATA
               STZ         $02
               PUSHLONG    LOCATION_OF_FPE+2  PUT FPE ADDRESS ON STACK
               PLA                            STORE FPE ADDRESS IN DIRECT
               STA         $00                PAGE
               PLA
               STA         $02
A1             MEMREGX     FMOVE,FP1,EXT_1    PUT DATA INTO FPE REGISTER 1
A2             MEMREGX     FADD,FP1,EXT_2     ADD SECOND VALUE TO REGISTER 1
A3             REGMEMX     FMOVE,FP1,ANS_1    RETRIEVE THE ANSWER IN EXTENDED
;                                             PRECISION FORMAT
               RTL
;
LOCATION_OF_FPE    DC      H'C0B0 0000'       ASSUME SLOT 3
;
;              FLOATING POINT EXTENDED DATA AREA
EXT_1          DC      H'0000 0000 0000 8000 3FFF'      VALUE=1.0
EXT_2          DC      H'0000 0000 0000 8000 3FFF'
;
;              RESULT SHOULD BE '0000 0000 0000 8000 4000' OR 2.0
ANS_1          DS      10
               END
```

8. About the MC68881 and SANE

The information in this chapter is excerpted from the "Apple Numerics Manual, Second Edition" chapters 27, 28, and 29. While all the information in the SANE manual may pertain to the operation of the MC68881 in the Macintosh II, the data here pertains only to the operation of the FPE when called by the FPE toolset.

Functions the same on both MC68881 and FPE software and SANE

The MC68881 and the FPE toolset return identical results for the following operations:

- addition
- subtraction
- multiplication
- division
- square root
- remainder
- round-to-integral value
- conversions between floating point formats
- negate
- absolute value.

Functions similar

For transcendental operations, the FPE gets results slightly less accurate than those returned by SANE; for some operations, the FPE gets different results for cases involving zero, Infinities, and NaNs.

The FPE returns slightly less accurate results than those returned by SANE in the following cases:

- binary scale (FPE truncates scale factors to 14 bits)
- base-e logarithm
- base-2 logarithm
- base-e logarithm of 1 + x
- base-e exponential
- base-2 exponential
- base-e exponential minus 1
- sine, cosine, tangent, arctangent
- integer exponentiation
- general exponentiation
- base-2 logarithm of 1 + x
- base-2 exponentiation minus 1
- compound interest
- annuity factor.

The FPE returns results with the same accuracy but behaves differently for zero, denormalized numbers, Infinities, and NANs:

- round-to-integer (when out-of-range the FPE preserves the sign)
- truncate-to-integer (when out-of-range the FPE preserves the sign)
- binary logarithm (same results except for 0 and Infinity).

All remaining operations available from SANE can be assumed to be as accurate and operate in the same manner for calls to the FPE toolset.

Accuracy of the MC68881's elementary functions

For the elementary functions, both the SANE and the FPE (MC68881) packages have errors in the least significant bits of the fraction part of the extended format results, but the SANE package errors rarely exceed the last bit, whereas the FPE errors can extend to as many as five bits. Hence, for individual elementary functions, both packages return results nearly identical when rounded to single or double precision. For complicated expressions involving elementary functions, the FPE is likelier to return an error in double precision results than the SANE packages are.

Controlling the environment

The FPE toolset converts the standard SANE environmental control calls to those needed by the MC68881.

Halts and Traps

The FPE toolset handles halts in the same way that the SANE package does.

Traps are not supported.

9. Programming Hints

1. If the FPE returns $0d1d in the response register, then the attempted operation was invalid. The only way to recover, short of powering off the system, is to call SANEReset from the toolbox or to use the following code:

```
lda     #0
sta     FPE_restore          (base register + 6)
lda     FPE_restore
```

Note that this is a 16 bit operation. If you are using a 6502/65C02-based system, you must do two 8-bit writes and two 8-bit reads.

2. When using the FPE Toolset from Pascal, C, or Basic, save intermediate results in the extended format. Use of other formats forces the compiler to convert your data values to and from extended, operations which will increase the execution time of your programs.

3. Whenever possible, store intermediate results in the FPE. Register-to-register operations can provide more than 10 times the performance of memory-to-register operations.

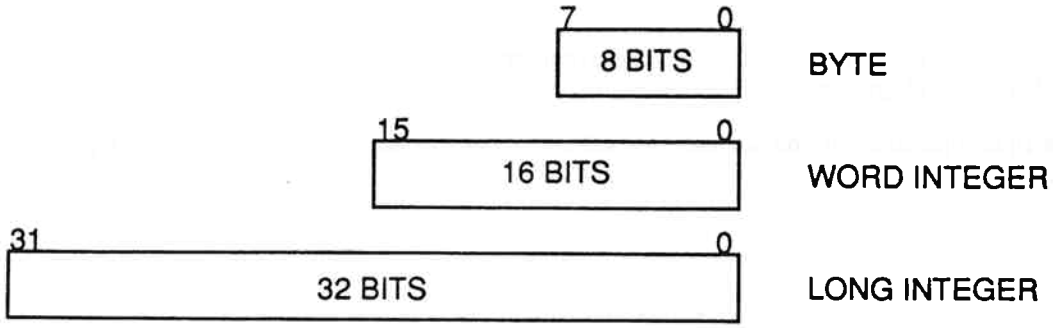
4. The FPE contains four (4) ID bytes which conform to the Apple standard. These bytes and their locations are:

Location	Value
$(00)cx05	$38
$(00)cx07	$18
$(00)cx0b	$01
$(00)cx0c	$af

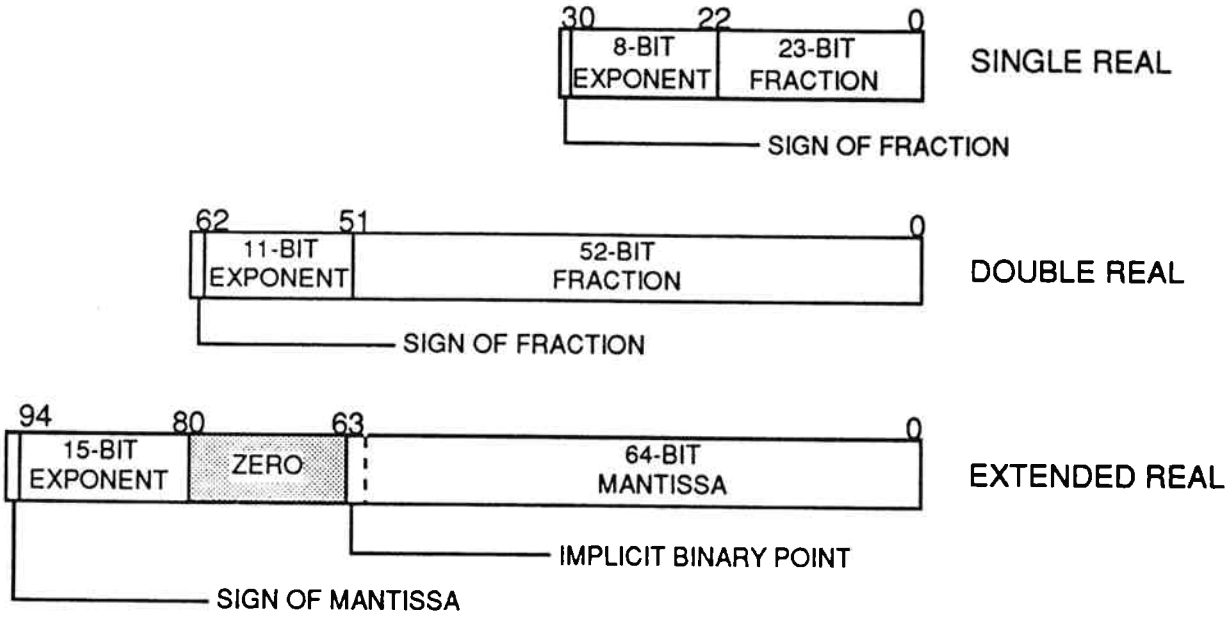
where x = the slot number.

Before reading the data in these locations, slotROM must be enabled by writing a value to $(00)c00b. Once done, the slot ROM must be disabled by writing a value to $c00a. Note that all accesses to the values should be done with the computer in 8-bit (short) index or accumulator mode.

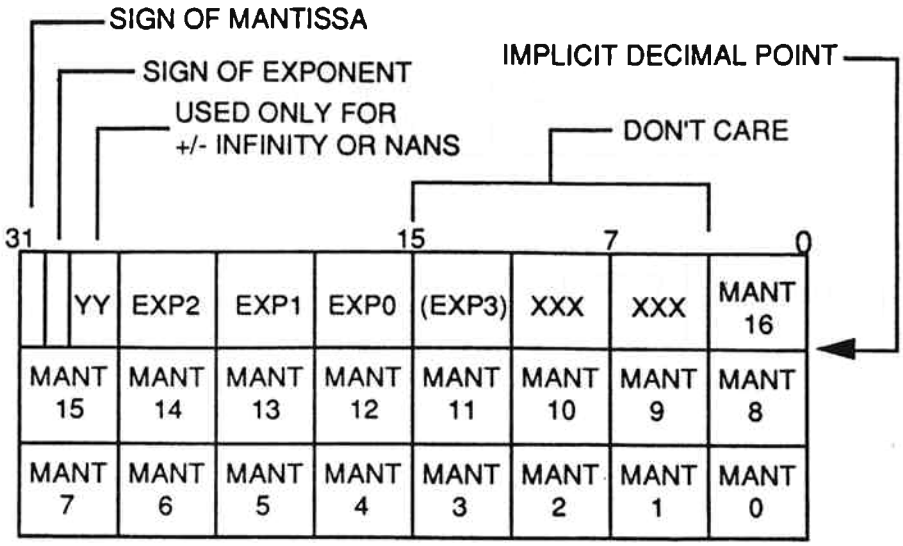
MC68881/68882 SIGNED INTEGER DATA FORMATS



MC68881/68882 REAL DATA FORMATS



MC68881/68882 PACKED BCD FORMAT



MANTn is the nth digit of the mantissa.
EXPn is the nth digit of the exponent. EXP3 is only generated during a move out operation if the source operand exponent exceeds the magnitude of a three digit exponent; otherwise it is a don't care. Only EXP0-EXP2 are used for input.
XXX are don't care bits, which are zero and ignored when read.

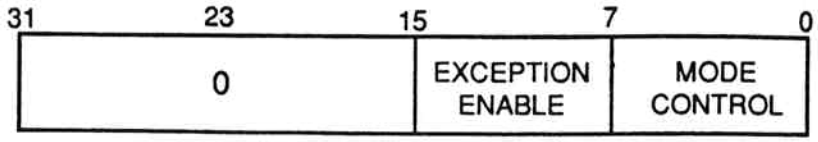
Operand Type	Word 5					Word 4	Word 0-3
	15	14	13	12	11...0	15...0	16-Digit Fraction
	SM	SE	Y	Y	3-Digit Exponent	1-Digit Integer	
+/- INFINITY	0/1	1	1	x	$FFF	$xxx	$00...00
+/- NAN	0/1	1	1	x	$FFF	$xxxx	Non-Zero, note 1
+/- SNAN	0/1	1	1	x	$FFF	$xxxx	Non-Zero, note 1
+/- ZERO	0	0/1	x	x	$000-$999	$xxx0	$00...00
- ZERO	1	0/1	x	x	$000-$999	$xxx0	$00...00
+ In-Range	0	0/1	x	x	$000-$999	$xxx0-$xxx9	$00...01-$99..$99
- In-Range	1	0/1	x	x	$000-$999	$xxx0-$xxx9	$00...01-$99..$99

Table A-1. Packed BCD String Definitions.

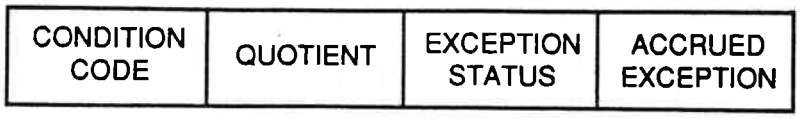
NOTES:

1. A decimal string with the SE and Y bits set, an exponent of $FFF, and a non-zero 16-digit decimal fraction is a NAN.
2. If a non-decimal digit ($A...$F) appears in the exponent of a zero, the number is converted to a true zero.

NON-DATA FLOATING POINT REGISTERS



FLOATING POINT CONTROL REGISTER



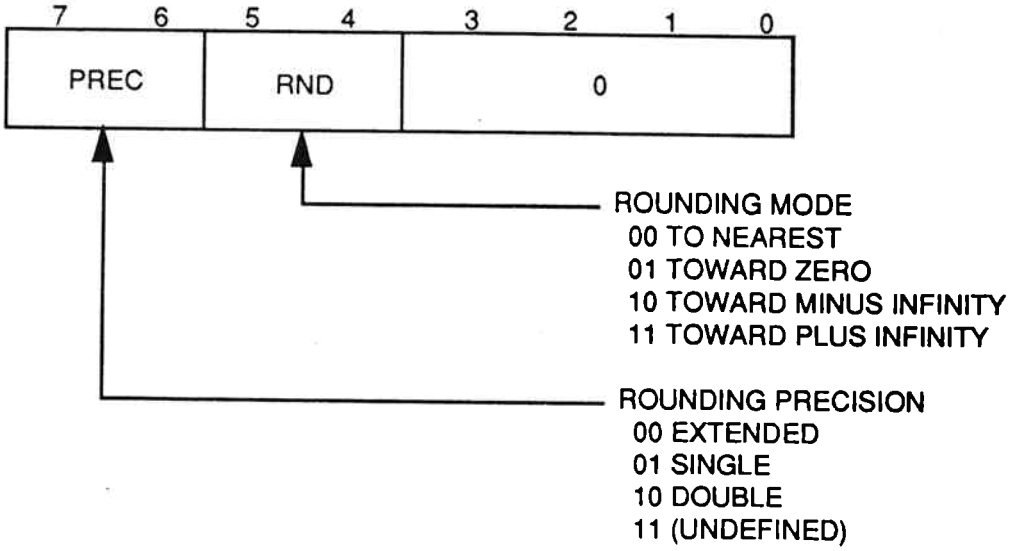
FLOATING POINT STATUS REGISTER

FLOATING POINT CONTROL REGISTER

FPCR EXCEPTION ENABLE BYTE

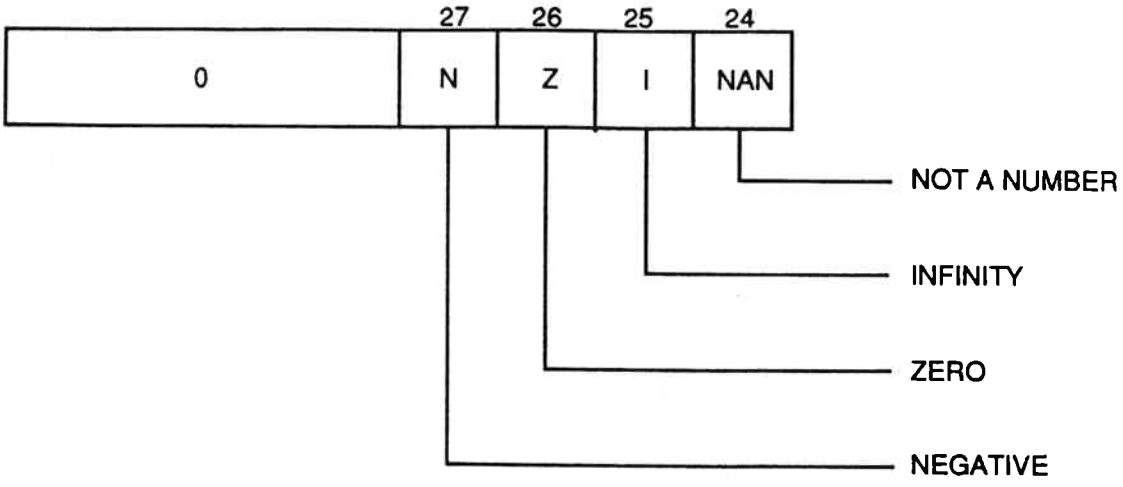
WARNING: DO NOT SET ANY BITS IN THIS BYTE!

FPCR MODE CONTROL BYTE



FLOATING POINT STATUS REGISTER

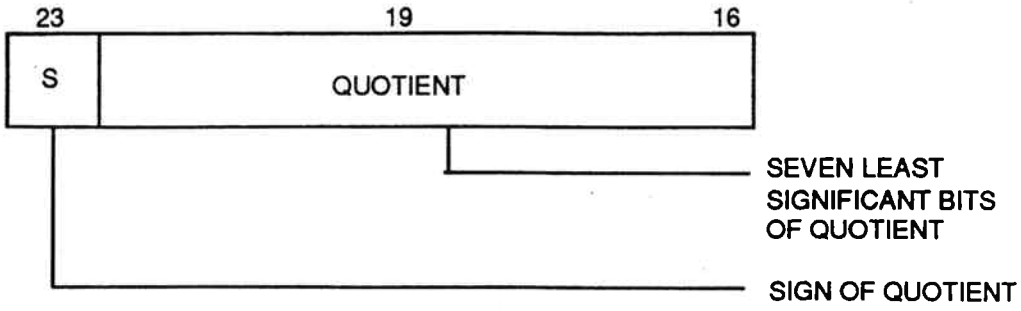
FPSR FLOATING POINT CONDITION BYTE



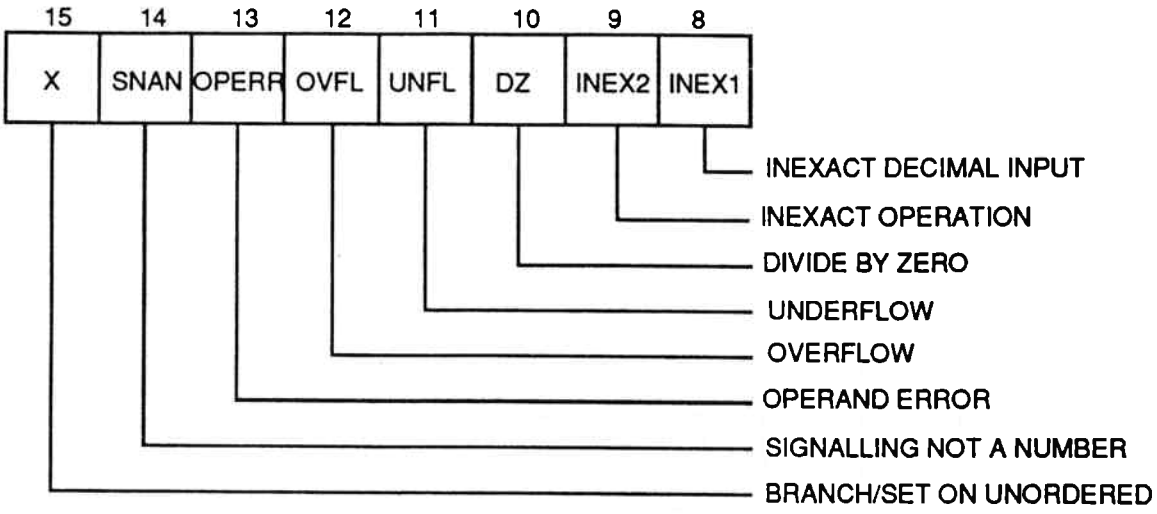
CONDITION CODE VERSUS RESULT DATA TYPE

N	Z	I	NAN	RESULT DATA TYPE
0	0	0	0	+ Normalized
1	0	0	0	- Normalized
0	1	0	0	+0
1	1	0	0	- 0
0	0	1	0	+ Infinity
1	0	1	0	- Infinity
0	1	0	1	+ NAN
1	1	0	1	- NAN

FPSR QUOTIENT BYTE



FPSR EXCEPTION STATUS BYTE



FPSR ACCRUED EXCEPTION BYTE