DECUS
PROGRAM LIBRARY

DECUS NO.	8-302
TITLE	OVERLAY MODIFICATIONS TO THE FLOATING-POINT SYSTEM PACKAGES, DEC-08-YQYA
AUTHOR	Peter L. Walton Submitted by: Dr. William S. Yamamoto
COMPANY	School of Medicine University of Pennsylvania Philadelphia, Pennsylvania
DATE	January 26, 1970
SOURCE LANGUAGE	MACRO-8

OVERLAY MODIFICATIONS TO THE FLOATING-POINT SYSTEM PACKAGES, DEC-08-YQYA

DECUS Program Library Write-up DECUS No. 8-302

ABSTRACT

These package overlays enable the expansion of the present basic package (DEC-08-YQYA) into 24 different packages each having unique specifications. Capabilities of the various packages include: basic and extended functions, and output controller, as in the basic package ; and output formatter with four I/O functions, extended memory referencing ability, and EAE utilization in multiplication and normalization subroutines. Included in each package is a correction of the error in the normalization routine in the basic package.

GENERAL NOTE

The packages presented ~~here~~ consist of modifications and additions to the 4 packages described in Floating-Point System Programming Manual, DEC-08-YQYA. The latter Manual is the basic reference for all packages and is altered or amended only as described below.

PACKAGE NAMING SCHEME

The naming scheme provides the easiest means for presenting 24 differenc packages. The general form for a package name is"FPPXYZ", where "FPP" denotes a floating-point system package, and "X", "Y", and "Z" represent the various symbols discussed below. A series consists of all packages having the same "X", "Y", or "Z", or combination of these.

"X" is either "S" or "E", where the FPPSYZ series includes 12 packages whose memory referencing abilities exist only in the single memory field containing the package; and the FPPEYZ series includes 12 packages whose memory referencing abilities exist in any (extended) memory field Ø-7, at least one of which contains the package.

"Y" is either "S" or "H", where the FPPXSZ series includes 12 packages whose arithmetic is carried out solely by "software" methods; and the FPPXHZ series includes 12 packages whose arithmetic is carried out in part by "hardware" methods, i.e. the EAE is employed.

"Z" is a single digit from 1 to 6, inclusive; the larger the digit, the more extensive the package's capabilities; where the FPPXY1 series includes 4 packages each having the basic functions (Note: The basic functions are FADD, FSUB, FMPY, FDIV, FGET, FPUT, FNOR, FEXT, SQUARE, SQROOT, INPUT, and OUTPUT.) as described in the Manual and on page 3; the FPPXY2 series includes 4 packages each having the basic functions (except OUTPUT) and the output controller as described in the Manual; the FPPXY3 series includes 4 packages each having the basic functions and the output formatter as described on page 5; the FPPXY4 series includes 4 packages each having the basic functions and the extended functions (Note: The extended functions are FSIN, FCOS, FATN, FLOG, FEXP, FLOAT, and FIX.) as described in the Manual and on page 7; the FPPXY5 series includes 4 packages each having the basic functions (except OUTPUT), the extended functions, and the output controller; and the FPPXY6 series includes 4 packages each having the basic functions, the extended functions, and the output formatter.

HARDWARE REQUIREMENTS

All packages require the PDP-8; in addition, those packages in the FPPEYZ series require a minimum of 8K of core memory to be useful, and those packages in the FPPXHZ series require the Extended Arithmetic Element to function.

SOFTWARE REQUIREMENTS

Since the packages are overlays to the basic packages DEC-08-YQYA, the latter 4 packages are "required."

CORE LIMITS

Each package as it is supplied on binary tape is intended to be loaded into Field Ø; however, any of the packages can be located in any memory field Ø-7 (see page 8). Regardless of the field, the core limits for the various packages remain the same and are:

FPPSS1	or	FPPSH1:	7; 4Ø-61; 56ØØ-7577
FPPES1	or	FPPEH1:	7; 4Ø-61; 63; 5553-7577
FPPSS2	or	FPPSH2:	7; 4Ø-62; 54ØØ-7577
FPPES2	or	FPPEH2:	7; 4Ø-63; 5353-7577
FPPSS3	or	FPPSH3:	7; 4Ø-62; 52ØØ-7577
FPPES3	or	FPPEH3:	7; 4Ø-63; 5153-7577
FPPSS4	or	FPPSH4:	7; 4Ø-61; 4757-7577
FPPES4	or	FPPEH4:	7; 4Ø-61; 63; 4732-7577
FPPSS5	or	FPPSH5:	7; 4Ø-62; 4557-7577
FPPES5	or	FPPEH5:	7; 4Ø-63; 4532-7577
FPPSS6	or	FPPSH6:	7; 4Ø-62; 44ØØ-7577
FPPES6	or	FPPEH6:	7; 4Ø-63; 4353-7577

LOADING PROCEDURE

Having selected the package:

(a) Load the parent package;
(b) Overlay (a) with the binary tape of the selected package;
(c) If the selected package is in the FPPXHZ series, overlay (b) with the binary tape of the EAE modification.

ENTRY POINT

For all FPPXYZ packages the pseudo-instruction interpreter is entered by JMS I 7, which is an effective JMS 56ØØ. Since INPUT and OUTPUT are pseudo-instructions in most packages, entry at points 74ØØ or 72ØØ as described in the Manual is usually unnecessary but is available for use in all packages.

USE

As in the basic packages, once the FPPXYZ interpreter is entered by JMS I 7, it decodes and executes pseudo-instructions as they occur sequentially in memory commencing with the register immediately after the one containing JMS I 7, until FEXT = Ø is encountered; control is then transferred to the register immediately following the one containing FEXT. (See Manual for further description.) For explanatory purposes use of the packages is described by series.

The FPPXY1 series includes FPPSS1, FPPSH1, FPPES1, and FPPEH1. FPPSS1 and FPPSH1 are used much the same as their parent package. Capabilities of each consist of the basic functions. Codes and effects of the basic function pseudo-instructions are described in the Manual and in the following INPUT and OUTPUT descriptions.

INPUT = ØØ13 is the pseudo-instruction equivalent of JMS I 5 as described in the Manual. OUTPUT = ØØ14 is the pseudo-instruction equivalent of JMS I 6 as described in the Manual. Both INPUT and OUTPUT make use of the package's teletype output routine (...; TSF; JMP .-1; TLS; ...), and hence the TTO flag must be set prior to execution of either pseudo-instruction. Setting the TTO flag is usually done with a TLS command in an initialization segment of the user program. Generally, the TTO flag must be set prior to any I/O pseudo-instruction of any package, including: INPUT and OUTPUT of the basic functions; and formatted OUTPUT, TYPE, SPACE, and CRLF of the output formatter.

FPPES1 and FPPEH1 are used as FPPSS1 is with the important exception that any of FADD, FSUB FMPY, FDIV, FGET, and FPUT which indirectly references "floating-point register" Y_F (i.e., the 3 register sequence whose first register has address Y) operates on Y_F of Field N, where $Ø \leq N \leq 7$ was set by the most recently executed FCDF NØØ pseudo-instruction.

FCDF NØØ = ØN17, where $Ø \leq N \leq 7$, is the pseudo-instruction which changes the "floating-point data field" indicator to N. The floating-point data field (FDF) indicated is the memory field containing the floating-point registers upon which indirectly referencing pseudo-instructions operate. As an example consider

```
      ...
      JMS I 7
      FCDF 4ØØ
      ...
      FGET I W
      FADD X
      FCDF 1ØØ
      FMPY I Y
      FCDF 3ØØ
      FPUT I Z
      ...
W,A
X,2
      24ØØ
      Ø
Y,B
Z,C
```

The latter coding gets the value in A_F of Field 4, adds to it 2.5_{10}, multiplies the sum by the value in B_F of Field 1, and puts the result in C_F of Field 3.

Summary of Facts for FPPEYZ Series Packages

1. The FDF can be referenced only by indirect addressing by pseudo-instructions FADD, FSUB, FMPY, FDIV, FGET, and FPUT (unless the FDF is the user program's instruction field, in which case Page Ø and the page containing the pseudo-instruction can be directly referenced).

2. Indirect addressing by a pseudo-instruction references the FDF indicated by the most recently executed FCDF NØØ pseudo-instruction. Thus one must keep track of what the FDF is while programming in the floating-point system; also the FDF is usually initialized at the outset of a user program.

3. The pseudo-instruction FCDF NØØ is analogous to the PDP-8 language instruction CDF NØ in all respects. Direct addressing by pseudo-instructions references same page or Page Ø floating-point registers; directly and indirectly referencing pseudo-instructions may be freely intermixed, along with all op code Ø pseudo-instructions since the FDF is temporarily suspended in favor of the package's instruction field during the execution of all op code Ø pseudo-instructions except FEXT. With the execution of FEXT, control is passed from the interpreter to the register following the one containing FEXT, and the FDF is left unchanged from what it was immediately prior to execution of FEXT. (Note: Op code Ø pseudo-instructions include all those whose codes have ØØØ in Bits Ø-2; i.e., all pseudo-instructions except FADD, FSUB, FMPY, FDIV, FGET, FPUT, and FNOR.)

4. The setting of the FDF in no way affects the operation of any machine-language instructions in the user program. The interpreter is exited with the (hardware) data field register the same as when the interpreter was entered; namely, the (hardware) data field is the instruction field of the package.

5. Register 63 gives the FDF, where C(63) = 62N1, and Field N is the FDF.

6. The contents of register 16 are lost when any pseudo-instruction except FEXT, FIX, TYPE, SPACE, CRLF, or FCDF is executed.

7. The package can be located in any memory field Ø-7, inclusive, by making minor changes (see page 8).

The FPPXY2 series includes FPPSS2, FPPSH2, FPPES2, and FPPEH2. FPPSS2 and FPPSH2 are used much the same as their parent package. Capabilities of each consist of the basic functions except OUTPUT (see Manual and page 3), and the output controller (see Manual). (Since immediately prior to an output command to the output controller both Register 62 and the accumulator must be set to give the format, an OUTPUT pseudo-instruction would be of limited value where the output controller is used. The interpreter should be exited, format set, and an effective JMS 72ØØ executed.) FPPES2 and FPPEH2 are used as FPPSS2 is with the addition of extended memory referencing ability.

The FPPXY3 series includes FPPSS3, FPPSH3, FPPES3, and FPPEH3 . FPPSS3 and FPPSH3 are used much the same as FPPSS1 is used with the addition of output formatting within the package. Capabilities of each consist of the basic functions (see Manual and page 3), and the output formatter.

The output formatter is a second-level interpreter that enables the user to control output format and type text strings without exiting the floating-point package. The output formatter decodes and executes four I/O pseudo-instructions: formatted OUTPUT, TYPE, SPACE, and CRLF. (Note: The output formatter destroys the contents of Register 57.)

FORMATTED OUTPUT = ØØ14 is a two-word pseudo-instruction that allows for formatted decimal output of C(FAC) while still under interpreter control. The second word gives the format. The general form of the pseudo-instruction is:

```
...
JMS I 7
...
OUTPUT
1ØØ'T + R
...
```

where T, R are octal integers Ø≤T, R≤77, and T is the total number of digits to be output, and R is the number of digits to the right of the decimal point. (Neither the decimal point nor the sign is considered to be a digit.) If T=Ø, output is in the E format; if R=Ø, the decimal point and all digits to the right of it are not typed. The + sign can be suppressed by changing C(7327) from Ø253 to Ø24Ø. The TTO flag must be set prior to execution (see page 3). A carriage return-line feed is not typed after output of C(FAC). (To get a CR-LF after output, change C(72Ø7) from 72ØØ to 1Ø55 and set C(55)≠ Ø.) Execution of formatted OUTPUT destroys the contents of Register 15, as does the output controller. As an example suppose C(X_F)=/ØØØ4/254Ø/ØØØØ/; then:

```
...
JMS I 7
...
FGET X
OUTPUT
1ØØ4
...
```

produces + 1Ø.75ØØ; or

```
...
JMS I 7
...
FGET X
OUTPUT
Ø3ØØ
...
```

yields + 1Ø; or

```
...
JMS I 7
...
FGET X
OUTPUT
ØØ77
...
```

gives +Ø.1Ø75ØØE+Ø2. The interpreter then goes on to the pseudo-instruction following the format word.

TYPE NØØ = ØN34, where Ø<N<7, is a two-word pseudo-instruction which allows for output of stored character strings while still under interpreter control. The second word contains the location of the first character in the string, and N gives the memory field in which the string is located. (Note: For FPPEY3 and FPPEY6 series packages, N can be any field Ø<N<7; for all other packages, N must equal the field containing the package.) Characters to be output are read as their 6-bit trimmed ASCII codes, two characters per register, and in left-right sequence within a register. Output proceeds in the latter sequence and register by register until the string terminator code (= 37) is encountered; control is then returned to the package interpreter and the pseudo-instruction following the second word of TYPE NØØ is executed. Four special codes are available:

ØØ = carriage return-line feed combination
34 = carriage return only
36 = line feed only
37 = string terminator

(Note: For 6-bit ASCII codes, see Digital's Introduction to Programming, 1968, Appendix B, p. B-1.) The TTO flag must be set prior to execution (see page 3). For example with an FPPEY3 or FPPEY6 series package and

```
...
JMS I 7
...
TYPE 7ØØ
Ø5ØØ
...
FIELD 7
*5ØØ
ØØØØ
4Ø4Ø
Ø134
Ø236
Ø337
...
```

produces

↲
↲
B A
C

where A, B, C are typed in alphabetical order, and ↲ is the non-printing symbol for CR-LF.

SPACE = ØØ54 is a two-word pseudo-instruction which allows for output of a specified number of spaces while still under interpreter control. This number of spaces is equal to the contents of the second word of the pseudo-instruction. If this is Ø, a CR-LF is output. The TTO flag must be set prior to execution (see page 3).

CRLF = ØØ74 is a one-word pseudo-instruction which allows for output of a carriage return-line feed combination while still under interpreter control. (The TTO flag must be set prior to execution.)

FPPES3 and FPPEH3 are used as FPPSS3 is with the addition of extended memory referencing ability.

The FPPXY4 series includes FPPSS4, FPPSH4, FPPES4, and FPPEH4. FPPSS4 and FPPSH4 are used much the same as their parent package. Capabilities of each consist of the basic functions, and the extended functions. Codes and effects of the extended function pseudo-instruction are described in the Manual and in FLOAT and FIX below.

FLOAT = ØØ11 is a pseudo-instruction that converts C(45) from an 11-bit, signed integer to its floating-point equivalent; the normalized result is left in the FAC.

FIX = ØØ12 is a pseudo-instruction that converts C(FAC) from a floating-point number to its 11-bit, signed integer equivalent, leaving the result in Register 45. Conversion truncates bits to the right of the binary point. If C(FAC) is not $-2\emptyset48<C(FAC)<2\emptyset48$, the resulting C(45) is in error.

FPPES4 and FPPEH4 are used as FPPSS4 is with the addition of extended memory referencing ability.

The FPPXY5 series includes FPPSS5, FPPSH5, FPPES5, and FPPEH5. FPPSS5 and FPPSH5 are used much the same as their parent package. Capabilities of each consist of the basic functions except OUTPUT, the extended functions, and the output controller. FPPES5 and FPPEH5 are used as FPPSS5 is with the addition of extended memory referencing ability.

The FPPXY6 series includes FPPSS6, FPPSH6, and FPPES6 and FPPEH6. FPPSS6 and FPPSH6 are used much the same as FPPSS4 is used with the addition of output formatting. Capabilities of each consist of the basic functions, the extended functions, and the output formatter. FPPES6 and FPPEH6 are used as FPPSS6 is with the addition of extended memory referencing ability.

List of pseudo-instructions presented here:

Mnemonic	Code	Comment
FNOP	ØØ1Ø	^{n}90-msec. delay; analogous to NOP
FLOAT	ØØ11	--
FIX	ØØ12	--
INPUT	ØØ13	--
OUTPUT	ØØ14	--
formatted OUTPUT	ØØ14	2-word pseudo-instruction
TYPE NØØ	ØN34	2-word pseudo-instruction
SPACE	ØØ54	2-word pseudo-instruction
CRLF	ØØ74	--
FCDF NØØ	ØN17	--

MEMORY RELOCATION OF PACKAGES

The binary tapes supplied contain the packages as intended for use in Field Ø. To relocate in another memory field $1 \le M \le 7$, first load selected package into Field Ø. Using an "up" bootstrap, transfer an image of the package to Field M. If the packate is of the FPPSYZ series, no changes need to be made. If, however, the package is in the FPPEYZ series, make the following alterations before using package: if the package's new field is M, change contents of

Register	From	To
5642	62Ø1	62M1
5653	62Ø1	62M1
5713	62Ø1	62M1
(TYPE+6	62Ø1	62M1 if FPPEY3 of FPPEY6 series)

NORMALIZATION SUBROUTINE CORRECTION

The normalization subroutine in DEC-08-YQYA packages can yield the illegal floating-point number /EXP/4ØØØ/ØØØØ/, which incidently is the only possible illegal number. If the result of an arithmetic operation indicated by a pseudo-instruction is the unnormalized, 4-word number /E/6ØØØ /ØØØØ/W/, where $1 \le W \le 3777$, and the format is the internally utilized triple-precision mantissa, then the normalized, 3-word result is /E-1/4ØØØ/ØØØØ/. This particular number appears to confuse the packages since it simultaneously represents -Ø and -2^{E-1} (and 2^{E-1}?). Given a result /E/6ØØØ/ØØØØ/W/, $1 \le W \le 3777$, normalization proceeds as follows:

(a) C(FAC) is negated to /E/1777/7777/$\tilde{W}$/, where $4001 \le \tilde{W} < 7777$ is the 2's complement of W; the result is left in the FAC. Negation is done so that only positive mantissas are normalized.

(b) C(FAC) is shifted left as a 36-bit word until Bit 1 is filled; and for each shift, the exponent is decreased by 1. Thus C(FAC) becomes /E-1/3777/7777/$2 \cdot \tilde{W}$, where $2 \le 2 \cdot \tilde{W} \le 7776$.

(c) Finally C(FAC) is negated again to regain the original negative value, and C(FAC) = /E-1/4ØØØ/ØØØØ/2·W/, where 2≤2·W≤7776. Any subsequent pseudo-instruction operation on C(FAC) either first clears Bits 24–35 of the mantissa or ignores their presence. Hence, the result is treated as /E-1/4ØØØ/ØØØØ/, which is the illegal number. The situation in which the problem arose involved the arithmetic calculation (-2.Ø) - (+2.Ø) during checkout computations of a machine-language version of the fast Fourier transform. The output result was -Ø.9999, which seemed incorrect at best. When the operands were examined, the following were found: X=/ØØØ2/6ØØØ/ØØØØ/, and Y=/ØØØ1/3777/7777/. The calculation proceeded as follows:

(a) Y is negated to -Y=/ØØØ1/4ØØØ/ØØØ1/ØØØØ/.

(b) -Y is scaled right until its exponent matches that of X; -Y=/ØØØ2/6ØØØ/ØØØØ/4ØØØ/.

(c) Both X and -Y are scaled right 1 bit to allow for overflow, then they are added: X= /ØØØ3/7ØØØ/ØØØØ/ØØØØ/, -Y=/ØØØ3/7ØØØ/ØØØØ/2ØØØ/, and X-Y=/ØØØ3/6ØØØ/ØØØØ/2ØØØ/.

(d) The result is normalized to X-Y=/ØØØ2/4ØØØ/ØØØØ/4ØØØ/, and thereafter treated as X-Y=/ØØØ2/4ØØØ/ØØØØ/, which is the illegal number. A listing of correction is given on page 10.

PROGRAM LISTINGS

Complete individual listings are not given; instead, for each package the modifications and where necessary their locations are mentioned. Modification listings are found on page 10.

FPPSS1: Parent listing is the same except for

6557	74ØØ	FLINTP
656Ø	72ØØ	FLOUTP

and the normalization correction.

FPPSS2: Parent listing is the same except for

6557	74ØØ	FLINTP

and the normalization correction.

FPPSS3: The FPPSS1 listing is the same except for

656Ø	52ØØ	OUTPUT

and the output formatter located in 52ØØ-5354.

FPPSS4: Parent listing is the same except for

6555	5563	FLOA
6556	4757	FIX
6557	74ØØ	FLINTP
656Ø	72ØØ	FLOUTP

and the normalization correction.

FPPSS5: Parent listing is the same except for

6555	5563	FLOA
6556	4557	FIX
6557	74ØØ	FLINTP

and the normalization correction.

FPPSS6: The FPPSS4 listing is the same except for

656Ø	44ØØ	OUTPUT

and the output formatter located in 44ØØ-4554.

FPPESZ series packages: The respective FPPSSZ series package listing is the same except for the extended memory modifications. The location of *FCDF is as follows:

FPPES1:	5553
FPPES2:	5353
FPPES3:	5153
FPPES4:	4732
FPPES5:	4532
FPPES6:	4353

FPPSHZ and FPPEHZ series packages: The respective FPPSSZ or FPPESZ series package listing is the same except for the EAE modifications.

```
                    /NORMALIZATION CORRECTION PATCH
                              *6647
                    /ROUTINE CHECKS FOR ILLEGAL FLOATING AC FOLLOWING
                    /NORMALIZATION OF NEGATIVE FAC
6647   5363                   JMP CHKNEG             /CALL TO CHECK ROUTINE
                              *6763
6763   4653         CHKNEG,   JMS I NEG              /NEGATE MANTISSA
6764   1045                   TAD HORD
6765   7104                   CLL RAL
6766   7640                   SZA CLA                /IS HORD=4000
6767   5600                   JMP I DNORM            /NO-EXIT
6770   1046                   TAD LORD               /YES
6771   7640                   SZA CLA                /IS LORD=0
6772   5600                   JMP I DNORM            /NO-EXIT
6773   1253                   TAD NEG                /YES
6774   3045                   DCA HORD               /6000=>HORD
6775   2044                   ISZ EXP                /EXP+1=>EXP
6776   7000                   NOP
6777   5600                   JMP I DNORM            /EXIT
                    /IN ORDER TO INSERT NORMALIZE CORRECTION ROUTINE,
                    /SUBROUTINE PRCHAR MUST BE INCORPORATED INTO SUB-
                    /ROUTINE INPUT, WHICH IS PRCHAR'S ONLY CALLER
                              *7151
7151   4772                   JMS I OUTPTT
                              *7155
7155   1374                   TAD MRBOUT
                              *7157
7157   5773                   JMP I RESTRT
7160   1375                   TAD MINCR
7161   7640                   SZA CLA
7162   5370                   JMP .+6
7163   1056                   TAD SWIT2
7164   7650                   SNA CLA
7165   5370                   JMP .+3
7166   1376                   TAD LFED
7167   4772                   JMS I OUTPTT
7170   1057                   TAD CHAR
7171   5742                   JMP I INPUT
7172   7344         OUTPTT,   OUT
7173   7401         RESTRT,   FLINTP+1
7174   7401         MRBOUT,   -377
7175   0162         MINCR,    377-215
7176   0212         LFED,     212
```

```
                *4400
                /OUTPUT FORMATTER; CALLED BY PSEUDO 00F4,
                /WHERE F=1,3,5,7
4400  0000 OUTPUT,    0
4401  1612            TAD I JUMPL          /GET PSEUDO
4402  7112            CLL RTR
4403  7012            RTR
4404  0213            AND MASK03           /GET BITS 6-7
4405  1214            TAD ACON7
4406  3057            DCA 57               /TABLE LOCATION
4407  1457            TAD I 57
4410  3057            DCA 57               /FUNCTION LOC.
4411  5457            JMP I 57             /JUMP TO FUNCTION
4412  5653 JUMPL,     JUMP
4413  0003 MASK03,    3
4414  4551 ACON7,     TABLE7
           /OUTPUT FUNCTION; CALLED BY PSEUDO 0014; WORD
           /FOLLOWING PSEUDO GIVES FORMAT AND HAS FORM T.100+R,
           /WHERE 0<=T,R<=77 AND T IS TOTAL NO. OF DIGITS TO
           /BE OUTPUT AND R IS NO. OF DIGITS TO RIGHT OF .
           /IF T=0, E FORMAT IS OUTPUT; IF R=0, NO . IS TYPED
4415  4231 OUTFNO,    JMS GETNXT           /GET FORMAT WORD
4416  0227            AND MASK77
4417  3047            DCA OVER2
4420  1443            TAD I OVER1
4421  4240            JMS ROTAT6
4422  0227            AND MASK77
4423  3062            DCA 62               /T=> 62
4424  1047            TAD OVER2            /R=>AC
4425  4630            JMS I FLOUTL         /OUTPUT C(FAC)
4426  5600            JMP I OUTPUT         /EXIT
4427  0077 MASK77,    77
4430  7200 FLOUTL,    FLOUTP
           /SUBROUTINE TO GET WORD FOLLOWING PSEUDO
4431  0000 GETNXT,    0
4432  1637            TAD I GO2LOC
4433  3043            DCA OVER1
4434  2637            ISZ I GO2LOC         /POINTER TO NEXT PSEUDO
4435  1443            TAD I OVER1
4436  5631            JMP I GETNXT         /EXIT WITH WORD IN AC
4437  5655 GO2LOC,    GO2
           /SUBROUTINE TO ROTATE AC 6 BITS RIGHT
4440  0000 ROTAT6,    0
4441  7012            RTR
4442  7012            RTR
4443  7012            RTR
4444  5640            JMP I ROTAT6
           /SPACE FUNCTION; CALLED BY PSEUDO 0054; WORD
           /FOLLOWING PSEUDO GIVES NO. OF SPACES TO BE OUTPUT;
           /IF 0, OUTPUT CR-LF
4445  4231 SPACE,     JMS GETNXT           /GET NO. OF SPACES
4446  7450            SNA                  /IS IT 0
4447  5261            JMP CRLF             /YES
4450  7041            CIA                  /NO
4451  3043            DCA OVER1            /NO. OF SPACES
4452  1257            TAD C240
```

```
4453  4660           JMS I OUTLOC         /OUTPUT SPACE
4454  2043           ISZ OVER1            /ARE WE DONE
4455  5252           JMP .-3              /NO-NEXT
4456  5600           JMP I OUTPUT         /YES-EXIT
4457  0240  C240,    240
4460  7344  OUTLOC,  OUT
            /CRLF FUNCTION; CALLED BY PSEUDO 0074
4461  1200  CRLF,    TAD OUTPUT
4462  3630           DCA I FLOUTL         /SET RETURN
4463  5664           JMP I FL7212         /DO CR-LF,EXIT
4464  7212  FL7212,  7212
            /TYPE FUNCTION; CALLED BY PSEUDO 0034; WORD FOLLOWING
            /PSEUDO CONTAINS STARTING ADDRESS OF CHAR. STRING;
            /CHAR.S ARE READ LEFT-RIGHT AS 6-BIT TRIMMED ASCII,
            /2 PER WORD, UNTIL HALF-WORD=37 IS READ, WHICH IS
            /STRING TERMINATOR; ALSO 00=CR-LF, 34=CR, 36=LF.
4465  7000  TYPE,    NOP
4466  7000           NOP
4467  4231           JMS GETNXT           /GET STRING SA
4470  3043           DCA OVER1            /STRING POINTER
4471  7000  TYPSWT,  NOP
4472  1443           TAD I OVER1          /GET WORD
4473  7000           NOP
4474  3047           DCA OVER2            /STORE IT
4475  7410           SKP
4476  7001  TYPRGT,  IAC
4477  7040           CMA
4500  3042           DCA AC1L             /-2 IF R; -1 IF L
4501  1047           TAD OVER2
4502  2042           ISZ AC1L             /IS L OR R NEXT
4503  5305           JMP .+2              /R
4504  4240           JMS ROTAT6           /L
4505  0227           AND MASK77           /GET 6-BIT CODE
4506  7450           SNA                  /IS IT 00
4507  5332           JMP TYPRLF           /YES-CR-LF
4510  1341           TAD M37              /NO
4511  7450           SNA                  /IS IT 37
4512  5600           JMP I OUTPUT         /YES-EXIT
4513  7001           IAC                  /NO
4514  7450           SNA                  /IS IT 36
4515  5337           JMP TYPLF            /YES-LF
4516  1342           TAD C2               /NO
4517  7450           SNA                  /IS IT 34
4520  5333           JMP TYPCR            /YES-CR
4521  1343           TAD M3               /NO
4522  7510           SPA
4523  1344           TAD C100
4524  1345           TAD C237             /GET 8-BIT CODE
4525  4660           JMS I OUTLOC         /OUTPUT CHAR
4526  2042  TYPCTL,  ISZ AC1L             /WAS L OR R JUST DONE
4527  5276           JMP TYPRGT           /L-DO R
4530  2043           ISZ OVER1            /R-ADVANCE POINTER
4531  5271           JMP TYPSWT           /NEXT WORD
4532  7100  TYPRLF,  CLL                  /0=>L FOR CR-LF
4533  1346  TYPCR,   TAD C215
4534  4660           JMS I OUTLOC         /OUTPUT CR
```

```
4535  7430                  SZL                         /WAS CR-LF CALLED FOR
4536  5326                  JMP TYPCTL                  /NO-NEXT CHAR
4537  1347  TYPLF,          TAD C212                    /YES-OUTPUT LF
4540  5325                  JMP TYPCTL-1                /NEXT CHAR
4541  7741  M37,            -37
4542  0002  C2,             2
4543  7775  M3,             -3
4544  0100  C100,           100
4545  0237  C237,           237
4546  0215  C215,           215
4547  0212  C212,           212
4550  0000                  0
            /TABLE FOR OUTPUT-CLASS FUNCTION LOOKUP
4551  4415  TABLE7,         OUTFNO
4552  4465                  TYPE
4553  4445                  SPACE
4554  4461                  CRLF
```

```
            /TYPE FUNCTION FOR EXTENDED MEMORY VERSIONS; CALLED BY
            /PSEUDO 0N34, WHERE N IS THE FIELD IN WHICH THE CHAR.
            /STRING IS LOCATED AND WORD FOLLOWING PSEUDO IS
            /STARTING ADDRESS OF STRING; CHAR.S ARE READ LEFT-
            /RIGHT AS 6-BIT TRIMMED ASCII, 2 PER WORD, UNTIL
            /HALF-WORD=37 IS READ, WHICH IS STRING TERMINATOR;
            /ALSO 00=CR-LF, 34=CR, 36=LF
                            *4465
4465  4750  TYPE,           JMS I SETFLL                /GET FIELD N
4466  3271                  DCA TYPSWT                  /SET SWITCH
                            *4473
4473  6201                  CDF 0
                            *4550
4550  4357  SETFLL,         SETFLD
```

```
                /EXTENDED MEMORY MODIFICATIONS
                        *63
0063    0000    FFIELD, 0
                        *4353
                /FCDF FUNCTION SUBROUTINE; CALLED BY PSEUDO 0N17,
                /WHERE N=NEW FLOATING DATA FIELD
4353    0000    FCDF,   0
4354    4357            JMS SETFLD              /FORM 62N1,PUT IN AC
4355    3063            DCA FFIELD              /SET FIELD INDICATOR
4356    5753            JMP I FCDF              /EXIT
                /SUBROUTINE TO TRANSFORM XNXX TO 00N0,ADD IN 6201
4357    0000    SETFLD, 0
4360    1766            TAD I JUMPL2            /GET PSEUDO
4361    7112            CLL RTR
4362    7010            RAR
4363    0367            AND MASK70              /GET N
4364    1370            TAD CCDF
4365    5757            JMP I SETFLD            /EXIT WITH 62N1 IN AC
4366    5655    JUMPL2, JUMP
4367    0070    MASK70, 70
4370    6201    CCDF,   6201
                /TABLE FOR PSEUDO-INSTRUCTION LOOKUP
4371    5716    TABLE1, FLAD
4372    5715            FLSU
4373    5762            FLMY
4374    6305            FLDV
4375    5676            FLGT
4376    5705    FLPTL,  FLPUT
4377    5735            NORF
                        *5600
                /MAIN INTERPRETER FOR EXTENDED MEMORY FLOATING-POINT
                /PACKAGE; FOLLOWING THE PSEUDO FCDF N00, ANY OF THE
                /PSEUDOS FADD,FSUB,FMPY,FDIV,FGET, OR FPUT WHICH
                /REFERENCE THE "FLOATING REGISTER" Y INDIRECTLY,
                /OPERATES ON Y OF FIELD N
5600    0000    FPNT,   0
5601    7300            CLA CLL
5602    3043            DCA OVER1               /CLEAR THIRD WORDS OF
5603    3047            DCA OVER2               /FAC AND OPERAND
5604    1600            TAD I FPNT              /GET PSEUDO
5605    3255            DCA JUMP
5606    2200            ISZ FPNT                /POINTER TO NEXT PSEUDO
5607    1255            TAD JUMP
5610    7106            CLL RTL
5611    7006            RTL
5612    0260            AND MASK3               /GET OP CODE(BITS 0-2)
5613    7450            SNA                     /IS OP CODE=0
5614    5337            JMP EXIT                /YES
5615    1374            TAD TABLE               /NO
5616    3256            DCA JUMP2               /LOCATION OF FUNCTION ADD.
5617    1255            TAD JUMP
5620    0263            AND PAGENO
5621    7650            SNA CLA                 /IS PAGE 0 REF. MADE
5622    5225            JMP .+3                 /YES
5623    1261            TAD MASK5               /NO
5624    0200            AND FPNT
```

```
5625  3257              DCA ADDR              /OPERAND ADD. BITS 0-4
5626  1262              TAD MASK7
5627  0255              AND JUMP              /OP. ADD. BITS 5-11
5630  1257              TAD ADDR
5631  3257              DCA ADDR              /ENTIRE ADD.
5632  1656              TAD I JUMP2
5633  3057              DCA 57                /FUNCTION ADD.
5634  1264              TAD INDRCT
5635  0255              AND JUMP
5636  7650              SNA CLA               /IS REF. INDIRECT
5637  5243              JMP LOOP01            /NO
5640  1657              TAD I ADDR            /YES
5641  3257              DCA ADDR              /EFF. ADD. OF OPERAND
5642  6201  DFSWIT,     6201                  /CDF N0, SET BY LAST FCDF
5643  7240  LOOP01,     STA
5644  1257              TAD ADDR
5645  3016              DCA 16                /ADD. OF OPERAND - 1
5646  1256              TAD JUMP2
5647  1265              TAD MFLPTL
5650  7510              SPA                   /IS PSEUDO FPUT OR FNOR
5651  5266              JMP GETOPR            /NO-GET OPERAND
5652  7640              SZA CLA               /YES-WHICH ONE
5653  6201  LOOP02,     6201                  /FNOR-RESET FIELD
5654  5457              JMP I 57              /JUMP TO FUNCTION
5655  0000  JUMP,       0
5656  0000  JUMP2,      0
5657  0000  ADDR,       0
5660  0017  MASK3,      17
5661  7600  MASK5,      7600
5662  0177  MASK7,      177
5663  0200  PAGENO,     200
5664  0400  INDRCT,     400
5665  3402  MFLPTL,     -FLPTL
            /ROUTINE TO PUT OPERAND IN FAC1
5666  7200  GETOPR,     CLA
5667  1416              TAD I 16
5670  3040              DCA EX1
5671  1416              TAD I 16
5672  3041              DCA AC1H
5673  1416              TAD I 16
5674  3042              DCA AC1L
5675  5253              JMP LOOP02            /RESET FIELD,JUMP TO FUNC.
                        *5705
            /FPUT FUNCTION; CALLED BY PSEUDO 6000+400.I+Y
5705  1044  FLPUT,      TAD EXP
5706  3416              DCA I 16
5707  1045              TAD HORD
5710  3416              DCA I 16
5711  1046              TAD LORD
5712  3416              DCA I 16
5713  6201              6201                  /RESET FIELD
5714  5201              JMP FPNT+1            /NEXT PSEUDO
            /FSUB FUNCTION; CALLED BY PSEUDO 2000+400.I+Y
5715  4775  FLSU,       JMS I OPMINS          /NEGATE FAC1,THEN ADD
            /FADD FUNCTION; CALLED BY PSEUDO 1000+400.I+Y
5716  4772  FLAD,       JMS I ALGN
```

```
                        *5720
5720  4773              JMS I UNORM
                        *5735
            /FNOR FUNCTION AND END OF FLAD; FNOR CALLED BY
            /PSEUDO 7000
5735  4771  NORF,       JMS I NORM
                        *5737
            /DISPATCHER FOR OP CODE 0 FUNCTIONS
5737  1255  EXIT,       TAD JUMP
5740  0260              AND MASK3           /GET PSEUDO BITS 8-11
5741  7450              SNA                 /IS IT FEXT(=0)
5742  5600              JMP I FPNT          /YES-EXIT PACKAGE
5743  1376              TAD ACON6           /NO
5744  3256              DCA JUMP2           /TABLE LOC. OF FUNC. ADD.
5745  1656              TAD I JUMP2
5746  3256              DCA JUMP2           /FUNCTION ADD.
5747  1200              TAD FPNT            /SAVE POINTER,ALLOWING
5750  3377              DCA GO2             /INT. CALL TO DEPTH OF 1
5751  1242              TAD DFSWIT          /INT. INDIRECTS REF. THIS
5752  3063              DCA FFIELD          /FIELD REGARDLESSLY
5753  3242              DCA DFSWIT          /SUSPEND INDIRECT FIELD
5754  4656              JMS I JUMP2         /CALL FUNC. AS SUBROUT.
5755  1377              TAD GO2
5756  3200              DCA FPNT            /RESTORE POINTER
5757  1063              TAD FFIELD
5760  3242              DCA DFSWIT          /RESTORE INDIRECT FIELD
5761  5201              JMP FPNT+1          /NEXT PSEUDO
            /FMPY FUNCTION; CALLED BY PSEUDO 3000+400.I+Y
5762  7201  FLMY,       CLA IAC
5763  1040              TAD EX1
5764  1044              TAD EXP
5765  3044              DCA EXP             /ADD EXPONENTS
5766  4770              JMS I MULT          /MULT. MANTISSAS
5767  5201              JMP FPNT+1          /NEXT PSEUDO
5770  6221  MULT,       DMULT
5771  6600  NORM,       DNORM
5772  6020  ALGN,       ALIGN
5773  6564  UNORM,      DUNORM
5774  4370  TABLE,      TABLE1-1
5775  6400  OPMINS,     MINUS2
5776  6544  ACON6,      TABLE6-1
5777  0000  GO2,        0
                        *6563
6563  4353              FCDF                /OP CODE 17
```

```
                /EAE PATCHES TO FPP SERIES
                          *6221
                /EAE MULTIPLY SUBROUTINE
6221  0000  DMULT,  0
6222  1363          TAD SMACLA
6223  3347          DCA SNSWIT
6224  4336          JMS SIGN
6225  1042          TAD AC1L
6226  3301          DCA PLIER
6227  1046          TAD LORD
6230  4277          JMS TIMES
6231  7710          SPA CLA
6232  7001          IAC
6233  1367          TAD HPROD
6234  3047          DCA OVER2
6235  1041          TAD AC1H
6236  3301          DCA PLIER
6237  1046          TAD LORD
6240  4277          JMS TIMES
6241  1047          TAD OVER2
6242  3047          DCA OVER2
6243  7004          RAL
6244  1367          TAD HPROD
6245  3370          DCA S2
6246  1045          TAD HORD
6247  3301          DCA PLIER
6250  1042          TAD AC1L
6251  4277          JMS TIMES
6252  1047          TAD OVER2
6253  3047          DCA OVER2
6254  7004          RAL
6255  1367          TAD HPROD
6256  1370          TAD S2
6257  3370          DCA S2
6260  1041          TAD AC1H
6261  3301          DCA PLIER
6262  1045          TAD HORD
6263  4277          JMS TIMES
6264  1370          TAD S2
6265  3046          DCA LORD
6266  7004          RAL
6267  1367          TAD HPROD
6270  3045          DCA HORD
6271  4760          JMS I NORMF
6272  7000          NOP
6273  2365          ISZ SGN
6274  5621          JMP I DMULT
6275  4773          JMS I MINS
6276  5621          JMP I DMULT
6277  0000  TIMES,  0
6300  7425          MQL MUY
6301  0000  PLIER,  0
6302  3367          DCA HPROD
6303  7501          MQA
6304  5677          JMP I TIMES
                    *6600
```

```
              /EAE NORMALIZE SUBROUTINE; NOTE: IN SINGLE CASE OF
              /C(FAC)=/E/6000/0000/7777>=W>=4000/, SUBROUTINE GIVES
              //E/6000/0000/ INSTEAD OF /E-1/4000/0001/; IE. BIT 23
              /LOSES SIGNIFICANCE IN THIS ONE INSTANCE
6600  0000  DNORM,      0
6601  7300              CLA CLL
6602  1046              TAD LORD
6603  7440              SZA
6604  5211              JMP NRMHL+1
6605  1045              TAD HORD
6606  7650              SNA CLA
6607  5240              JMP CKOVR2
6610  1046  NRMHL,      TAD LORD
6611  7421              MQL
6612  1045              TAD HORD
6613  7411              NMI
6614  3045              DCA HORD
6615  1047              TAD OVER2
6616  7421              MQL
6617  7441              SCA
6620  7450              SNA
6621  5600              JMP I DNORM
6622  1367              TAD M15
6623  7500              SMA
6624  5244              JMP NRMLO
6625  1370              TAD C14
6626  3231              DCA NOSHL
6627  1046              TAD LORD
6630  7413              SHL
6631  0000  NOSHL,      0
6632  3046              DCA LORD
6633  7501              MQA
6634  3047              DCA OVER2
6635  1231              TAD NOSHL
6636  7001              IAC
6637  5363              JMP FEXPON
6640  1047  CKOVR2,     TAD OVER2
6641  7450              SNA
6642  5365              JMP FEXPON+2
6643  7421              MQL
6644  7200  NRMLO,      CLA
6645  1046              TAD LORD
6646  7411              NMI
6647  3045              DCA HORD
6650  7501              MQA
6651  3046              DCA LORD
6652  3047              DCA OVER2
6653  7441              SCA
6654  1370              TAD C14
6655  5363              JMP FEXPON
                        *6763
6763  7041  FEXPON,     CIA
6764  1044              TAD EXP
6765  3044              DCA EXP
6766  5600              JMP I DNORM
6767  7763  M15,        -15
6770  0014  C14,        14
```