DECUS
PROGRAM LIBRARY

DECUS NO. 12-68

TITLE A PDP-8 FLOATING POINT SOFTWARE PACKAGE SIMULATOR USING A FPP-12 FLOATING POINT PROCESSOR

AUTHOR L. G. Boxall and R. H. Abel

COMPANY Colorado State University
Fort Collins, Colorado

DATE March 2, 1972

SOURCE LANGUAGE ASSEMBLER-PDP and FPP-12

A PDP-8 FLOATING POINT SOFTWARE PACKAGE SIMULATOR USING A FPP-12 FLOATING POINT PROCESSOR

DECUS Program Library Write-up DECUS NO. 12-68

INTRODUCTION

The recent introduction of the DEC FPP-12 Floating Point Processor has produced a serious dilemma for operators using well established program systems. In order that the speed of the FPP-12's floating point calculations can be utilized without first having to rewrite major computer programs, we have developed an FPP-12 simulator. The FPP-12 simulator will accept all of the instructions used in the PDP-8 Floating Point System Packages[1] as well as provide many additional features. The execution rate for the single instruction (i.e., FADD) is about 6 times faster using the FPP-12 simulator than the conventional software package, while the execution rate for functions (i.e., FCOS) is increased by about 10 times. The FPP-12 simulator utilizes less core than the conventional software package and, in addition, provides 48 bit precision in all of its floating point calculations.

DESCRIPTION OF THE SIMULATOR PACKAGE

The FPP-12 simulator consists of three separate program units based on modifications made to the PDP-8 Floating Point System and the FPP-12 Subroutine Library. The simulator must reside in field 0.

1. The Basic FPP-12 Simulator (FPPSIM). This unit is a binary program containing the system used to translate the software floating point instructions into a form compatible with the FPP-12 processor. The floating point input and output routines for the teletype along with several other commonly used routines are included in the basic package. A complete list of these routines is given in the description of the instruction set. The basic interpreter and the I/O Routines can be independently relocated in core by redefining the parameters "SIMUL" and "INOUT" in the source programs followed by assembly and combination of the two binary programs.

2. The FPP-12 Simulator Patch (PTCHSIM). This unit is a source program which adds to the FPPSIM program all the special functions required by programming or the current problem. The programmer may add any of the routines in the FPP-12 Subroutine Library contained in the PTCHSIM program as well as user-written PDP-8 and FPP-12 routines. The FPP-12 routines can be stored in any field of core, thus conserving lower core storage. A facility has been added which permits the variation of the number of digits following the decimal point in the E format output. The parameter "NDIGIT" is set to the number of required digits before assembling the PTCHSIM source program. Any division by zero errors are trapped by the FPP-12 processor. The error flag at location 61 is set and the floating point exponent (location 44) is set to its maximum value. A NOP instruction at location "ERR" has been left in the program so that additional corrective action can be taken by the programmer before the calculations are continued. PTCHSIM allows the programmer to insert a JMS instruction into the simulator package if such action is required. To avoid the illegal resetting of indicators within the simulator, the simulator must not be used in the correction subroutine.

3. The FPP-12 Simulator I/O Controller (IOCNTRL). This binary unit is added to the binary simulator package to provide formatted floating point output. The operation of the IOCNTRL program is the same as that described in the software floating point manual with one exception. The number of digits to the right of the decimal point is stored in core location 51 in lieu of the accumulator; this location is not changed during any of the floating point operations. The controller can be relocated by redefining the parameter "CNTRL" in the source program and reassembling. If the I/O routines have been relocated, the IOCNTRL program must be re-assembled with the new INOUT value in the source program. The contents of auto index 15 is lost during the controller's operation.

DESCRIPTION OF THE INSTRUCTION SET

The instructions have been divided into four classifications. The formats and restrictions for each class are listed below.

1. Enter simulator

   FENTER = JMS I 7 = 4407

   Restriction: must be in the same field as the simulator.

   Note: the current data field is saved on entering the simulator and then restored on leaving the simulator.

2. Exit simulator

   FEXT = 0000

   Restriction: must be in the same field as the simulator.

3. Memory reference

   FADD = 1000

   FSUB = 2000

   FMPY = 3000

   FDIV = 4000

   FGET = 5000

   FPUT = 6000

   FNOR = 7000

   Restriction: data and instructions must be in the same field as the simulator.

   Note: the combination of these commands with direct and indirect addressing is the same as that used in the software floating point package.

4. Functions

a. P-mode . These are subroutines containing PDP instructions which can be called from within a consecutive string of floating point instructions without calling FEXT first (i.e., INPUT of OUTPUT).

Restrictions

i. must start and end in P-mode and in the same field as the simulator;
ii. it can use only class 1, 2, 3 and 4-b instructions for its floating point calculations;
iii. FAC resides in core locations 44 to 66 in the same field as the simulator.

Instruction Code

$0040 + N \quad 1 \leq N \leq 17 \quad (N_8)$

N = 1 INPUT (reserved for simulator)

N = 2 OUTPUT (reserved for simulator)

Note: the address of the subroutine is entered into the Nth location of Table 8 by use of the FPP-12 Simulator Patch Program.

b. FPP-12 mode. These are subroutines containing only FPP-12 coding. These routines may be user-written programs or routines added from the modified FPP-12 library contained in the FPP-12 Simulator Patch Program. Since the traps have been removed from the FPP-12 library, care must be taken to insure that only division by zero errors can interrupt the FPP-12.

Instruction Code

$0F00 + M \quad 1 \leq M \leq 17 \quad (M_8)$

F = field containing FPP-12 routine

The following routines are provided in the Basic FPP-12 Simulator Package:

0001	SQUARE	FAC * FAC ——> FAC
0011	NEGATE	−FAC ——> FAC
0012	FIX	fix FAC ——> signed 45
0013	INPUT	TTY ——> FAC
0014	OUTPUT	FAC ——> TTY FAC destroyed
0015	FLOAT	float signed 45 ——> FAC
0016	FL024	float signed 45, 46 ——> FAC

The addresses of additional subroutines and the names of the required FPP-12 Library routines are entered into the Mth location of Table 2 by use of the FPP-12 Simulator Patch Program. The FPP-12 Base register and only the required functions are loaded into any field of core, thus saving on the program core requirements. The active parameter table and the FPP-12 index registers reside in the simulator's field at locations 37 to 46 and 6302 to 6311 (FPPSIM = 5600), respectively.

The trap handler and I/O handler have been eliminated from the standard FPP-12 Support Library. The following standard corrections have been inserted into the library in lieu of the error traps.

Trap No.	Correction Implemented
0	complement base and continue
1	set argument = 1 and continue
2	set argument = 1 and continue
3,6	complement argument and continue
4	set to largest positive number and exit if argument is positive; set to zero and exit if argument is negative
5, 11	set to zero and exit
7	set to largest negative number and exit
12, 10	set to largest negative number and exit

The cause for each trap is given in the FPP-12 Support Library User's Guide.[2]

ASSEMBLY OF FPP PACKAGE

The final binary floating point package is put together in the following sequence:

1. Modify the PTCHSIM program to fulfill program requirements. The documentation in the source program is self-explanatory. Assemble using the FPP assembler and save the binary file.

2. Zero the binary working area in the Dial-MS system and then add the FPPSIM binary file.

3. Add the saved PTCHSIM binary file.

4. Add the IOCNTRL binary file if required.

5. Save the FPP Floating Point Binary Package and use in lieu of the software package.

FPP-12 SIMULATOR CORE REQUIREMENTS

The core location parameters in the original simulator package are:

```
SIMUL = 5600
INOUT = 6400
CNTRL = 5400
```

and the core utilization in field 0 is as follows:

7	address of simulator entry point
37 - 46	FPP-12 Active parameter table
44 - 46	FAC
47	0F00, F = field of base register
50	Address of base register

51	IOCNTRL, number of digits right of decimal
55	7777, if 0, then no CR-LF after output
56	7777, if 0, then no LF after CR in INPUT
57	last character read in
60	equals 0 if no conversion took place
5400-5577	IOCNTRL (if loaded)
5600-6206	FPPSIM
6207-6301	FPP-12 Routines in basic package
6302-6311	index registers
7000-7577	Input-Output routines

TESTING SIMULATOR

The simulator has been extensively used here for two months with no problems. The software floating point package was replaced by the simulator in numerous programs ranging from real time data acquisition systems to multiple regression analysis programs. The results from the modified programs never varied by more than 1 in the fifth figure from the previous results. The execution time for a data analysis program containing about 80% floating point instructions decreased from about one hour to seven minutes by the use of the simulator. The increased computing speed of the simulator made it possible to use the PDP-12 exclusively (rather than the university's expensive central computing facility) in carrying out all the computations required for a publication on the electrochemical behavior of aluminum chloride salts.[3] The fact that computation was being done in the laboratory greatly facilitated the speed and ease of program development.

REFERENCES

1. PDP-8 Floating Point System Programming Manual, 8-5-S, Digital Equipment Corp., (1965).

2. FPP Support Library User's Guide, No. DEC-12-YEXA-D, Digital Equipment Corp., (1971).

3. L. G. Boxall, H. L. Jones and R. A. Osteryoung, submitted for publication in J. Phys. Chem. (1972).