ELLIOTT 903

Volume 2: PROGRAMMING INFORMATION

Part 2: PROGRAM DESCRIPTIONS

Section 8: QF (FLOATING POINT SUBROUTINES)

Contents

Chapter 1: DESCRIPTION

1. 1 INTRODUCTION

1. 1. 1 Purpose.

QF is used to perform operations on floating-point numbers.

1. 1. 2 Summary.

QF contains routines for operations corresponding to all the fixed-point operations except the function 15 (Functions 3 and 6 have special meanings in QF).

When entered, QF proceeds to interpret the instructions in the locations immediately following the entry-point in the user's program. Thus, operations are performed on floating-point numbers by placing in store the corresponding fixed-point instructions.

Two formats are available for floating-point numbers (see Paragraph 1. 2. 2).

1. 1. 3 Form of Distribution.

The program is distributed as an Elliott SIR mnemonic tape which has QF declared as a global identifier.

1. 1. 4 Method of Use.

See Paragraph 1. 1. 2. QF may be run in any program level.

1. 1. 5 Restrictions.

See Paragraph 1. 2. 4.

1. 1. 6 Accuracy.

If the result of the operation is y, multiplication gives a maximum error of $2^{-34}y$, division gives a maximum error of $2^{-32}y$. All other operations give a maximum error of $2^{-35}y$.

1. 2 FUNCTIONS

1. 2. 1 Notation.

x(man) = mantissa of floating-point number, x.

x(exp) = exponent of floating-point number, x.

x = a floating point number held in 2 or 3 words from location X.

f = the floating point number held in the floating point accumulator (FPA).

b = the contents of the pseudo B-register (FBREG)

s = the contents of the pseudo S-register.

C(x) = the contents of location X

:= means "becomes equal to"

1. 2. 2 Format.

One of two formats may be used to hold a floating-point number in store. Normally the packed format is used, but the unpacked format allows a wider range of numbers and slightly greater accuracy. The two formats are summarised in the table below. In the unpacked format, the mantissa is a double-length fraction held in two consecutive locations and the exponent is a single-length integer held in the next location. For the packed format, the mantissa is truncated and the exponent held in the seven least significant bits of the second store location. In this case the exponent must be in the range -64 to +63.

Format	Location	Bit 18	Bits 17-8	Bits 7 - 1
Packed	X	sign	most significant bits of mantissa	
	X+1	0	least sig. bits of mantissa	exponent
Unpacked	X	sign	most significant bits of mantissa	
	X+1	0	least significant bits of mantissa	
	X+2	← exponent →		

Negative number representation for exponent and mantissa is by the usual 2's complement notation.

All internal working of QF uses the unpacked format.

Examples of floating point numbers in the two formats:

Number	Locn	Packed	Unpacked
0. 25=	X	010 000 000 000 000 000	010 000 000 000 000 000
0.5×2^{-1}	X+1	000 000 000 001 111 111	000 000 000 000 000 000
	X+2	Not Used	111 111 111 111 111 111
$1-2^{-27} \times 2^{63}$	X	011 111 111 111 111 111	011 111 111 111 111 111
$\doteqdot 9.2 \times 10^{18}$	X+1	011 111 111 110 111 111	011 111 111 110 000 000
	X+2	Not Used	000 000 000 000 111 111
-1.0×2^{-64}	X	100 000 000 000 000 000	100 000 000 000 000 000
$\doteqdot -0.5 \times 10^{-21}$	X+1	000 000 000 001 000 000	000 000 000 000 000 000
	X+2	Not Used	111 111 111 111 000 000

1. 2. 3 Entry and Exit.

Entry is made by

11 QF
8 QF + 1 } to use packed format

or

11 QF
8 QF + 2 } to use unpacked format.

QF proceeds to interpret and execute the pseudo-program using

a pseudo-accumulator (FPA)
a pseudo-B-register (FBREG)
and a pseudo-S-register (QF)

See Paragraph 1. 2. 4 for the effects of each function.

Exit is made by placing +0 in the location after the last instruction to be interpreted. Control is then transferred to the location following the zero location. The Machine Accumulator and B register are not preserved. The pseudo-Accumulator and B registers are not affected by entry and exit.

1. 2. 4 Available Operations.

See Paragraph 1. 2. 1 for notation used.

Table 1

Pseudo Instruction.	Name	New Contents of		Remarks
		FPA	X	
0 X	Load pseudo B-register	f	x	b:=C(X) See Notes 1 & 3
1 X	Add	f+x	x	
2 X	Negate & Add	x-f	x	
3 X	Exchange	x	f	Not a basic 903 operation
4 X	Load FPA	x	x	
5 X	Store FPA	f	f	See Note 2
6 N	Conversion Routines	f	x	See Table 2
7 X	Jump if f= zero	f	x	See Note 3
8 X	Jump	f	x	See Note 3
9 X	Jump if f<0	f	x	See Note 3
10 X	Count in store	f	"x+1"	See Note 3
11 X	Store pseudo SCR	f	s	See Note 3
12 X	Multiply	f*x	x	
13 X	Divide	f/x	x	See Note 4
14 N	$*2^N$	$f*2^N$	x	N<4096 See Note 5
14 N	$*2^{N-8192}$	$f*2^{N-8192}$	x	N≥4096
15 N	error	f	x	See Paragraph 1. 3

Notes

(1) The instruction 0 0 is interpreted as a terminator for floating-point working (See Paragraph 1. 2. 3).

(2) If packed format is in force during interpretation of a 3 or 5 instruction, then a test is made whether

$$-64 \leq f(exp) < +64$$

If f(exp) < -64 then x:=0 and the next instruction is interpreted.

If f(exp) ≥ +64 then an error indication is output (See Paragraph 1. 3).

(3) - These instructions operate on single word items. The instructions 7, 8 & 9 may jump to another interpreted instruction: they must not jump out of the interpreted program except via a terminator.

(4) If an attempt is made to divide by zero an error indication is output (See Paragraph 1. 3).

(5) The results of the following instructions are always standardised:

1, 2, 12, 13, 14.

The instruction 14 0 may be used to standardise the contents of the FPA.

(6) Modified instructions may be used and, if they are, the contents of the pseudo B-register are added to the address digits before obeying an instruction.

(7) Literal addresses may not be used i. e. constants must be stored in the correct format by the user.

Table 2

The address of a 6 instruction determines its meaning.

Function	Effect
6 1	Instructions interpreted after this assume packed format.
6 2	" " " " " unpacked "
6 3	Convert a single-length integer to a floating-point number and place the result in the FPA. Location QF+3 contains the address where the integer is held.
6 4	Form the integral part of the number in the FPA. Location QF+4 contains the absolute address where the (single-length) integer is to be placed. This routine always rounds down.
6 5	Convert a fixed-point fraction to a floating-point number and place the result in the FPA. Location QF+5 contains the absolute address of the (single-length)fraction.
6 6	Convert the number in the FPA to a fixed-point fraction. Location QF+6 contains the absolute address where the (single-length) fraction is to be placed. This routine always rounds down.

The instructions 6 1 and 6 2 do not convert numbers; they define the action of following instructions:-

In 6 3 to 6 6 the arguments and the addresses in QF+3 to QF+6 are unaffected. Error indications are output if an impermissible address is used or if overflow occurs. (See Paragraph 1. 3).

Example

An integer is held in INT1 and a real number in RL2

The following section of program places the floating-point form of the first in RL1 and the entier of the second in INT2.

```
(SET ADDRESSES IN QF WORKSPACE)
    4     PSI1          (PSI1 holds the address of INT1)
    5     QF+3
    4     PSI2          (PSI2 holds the address of INT2)
    5     QF+4

(NOW PERFORM CONVERSIONS)
    11    QF            (ENTER QF)
    8     QF+1
    6     3             (INT1 in the FPA)
    5     RL1
    4     RL2
    6     4             (RL2 stored as integer)
    +0                  (RETURN to normal working)
    8     ;+0

PSI1  0     INT1
PSI2  0     INT2
```

1. 3 ERROR INDICATIONS

1. 3. 1 Standard Indication

If an error occurs a message is displayed output on the teleprinter followed by 5 inches of blank tape on the punch. Recovery can be made and the program continued by entry at 9.

1. 3. 2 Errors Detected.

Message	Significance	Effect of re-entry at 9
QF!	Impermissible instruction	Dynamic stop at 9
ROF!	Floating-point over-flow (5 or 13 instruction)	Result is taken as largest number of correct sign that can be held in packed format.
RTI!	Integer overflow	Result is taken as largest integer of correct sign.
RTF!	Fraction overflow (6 6 instruction)	Result is taken as largest fraction of correct sign.

1.4 METHOD USED

The following steps are carried out for each pseudo-instruction interpreted.

(1) The pseudo S-register is incremented.

(2) The function and the address (modified if required) bits of the interpreted instruction are stored.

(3) Control is transferred to the appropriate routine to execute the instruction.

(4) Control is returned to the interpreter (via a standardising routine for instructions 1, 2, 12, 13, 14).

Subroutines from 903/Algol have been used for all arithmetic operations.

1.5 STORE USED

The floating-point package occupies approximately ~~900~~ 750 locations.

1.6 TIME TAKEN

The following times are approximate:-

Function Number	Operation	Times in Microseconds
0	Set pseudo B-register	440
1	Add	2150 to 3850 (average 2500- see Note 1)
2	Negate and Add	2480 to 4180 (average 2800- see Note 1)
3	Exchange	1980 packed or 1810 unpacked
4	Load FPA	770
5	Store FPA	770
6	Specifies format for the following instructions	440
7	Jump if zero	440
8	Jump	440
9	Jump if negative	440
10	Count in store	400
11	Store pseudo SCR	440
12	Multiply	2140
13	Divide	~~9000~~ 4300 to ~~10700~~ 6900 (average ~~9400~~ 4700 see Note 1)
14	Shift	810 to 2510 (average 1210- see Note 1)
	ENTRY	50
0 0	EXIT	150

Notes

(1) The time depends on the number of places the mantissa is shifted to standardise the result of the operation. The average given assumes a shift of 4 places.

(2) For modified instructions add 125 μs to the time taken.