October 4, 1963
Revised October 17, 1963.
Revised October 23, 1963

FP6000 Project Memorandum #29

Notes on the Order Code

Revisions have been underlined.

W. R. Whittall
Technical Services
Data Systems Department.

DISTRIBUTION LIST

L. R. Wood

B. Z. de Ferranti

A. Jackson
G. Calverley

Engineering

G. Collins
E. Strain
B. Ovenell
H. Reddering
B. Cellier
K. Bower
J. Davidson

Technical Services

L. Albert
J. D. Buck
J. Chapman
L. A. Cragg
A. G. Henry
L. F. Lawry
F. M. Longstaff
R. F. Johnston
M. J. Marcotty
R. Moore
E. McDorman
J. W. McSherry
D. E. Nuttall
D. K. Ritchie
R. Sadana
I. P. Sharp
A. Sharp
D. K. Smith
V. Taylor
W. R. Whittall
File
B. Daly
P. Boyd

October 4, 1963
Revised October 17, 1963.
Revised October 23, 1963

FP6000 Project Memorandum #29

Notes on the Order Code

Revisions have been underlined.

W. R. Whittall
Technical Services
Data Systems Department.

DISTRIBUTION LIST

L. R. Wood

B. Z. de Ferranti

A. Jackson
G. Calverley

Engineering

G. Collins
E. Strain
B. Ovenell
H. Reddering
B. Cellier
K. Bower
J. Davidson

Technical Services

L. Albert
J. D. Buck
J. Chapman
L. A. Cragg
A. G. Henry
L. F. Lawry
F. M. Longstaff
R. F. Johnston
M. J. Marcotty
R. Moore
E. McDorman
J. W. McSherry
D. E. Nuttall
D. K. Ritchie
R. Sadana
I. P. Sharp
A. Sharp
D. K. Smith
V. Taylor
W. R. Whittall
File
B. Daly
P. Boyd

NOTES ON THE FP6000 ORDER CODE

1. FUNCTIONS 004, 014, 104)
005, 015, 105) (additions)

These functions first perform the specified addition on the 24 bit signed operands, set carry if the result is negative and then clear the sign bit of the result before storing.

2. FUNCTIONS 006, 016, 106)
007, 017, 107) (subtractions)

These functions first perform the specified subtraction on the 24 bit operands, set carry if there is an overall borrow (treating the sign bits as having positive weight) and then clear the sign bit of the result before storing.

NOTE Function 027 is identical to function 007 except that the value of x is left unchanged.

3. MULTIPLICATION 040, 041, 042

(a) Overflow can only be set when n = x = -1 and result is then -1

(b) In the 042 instruction x* may be positive or negative.

(c) 041 CAN SET OVERFLOW IF ROUNDING OVERFLOWS

4. MULTIPLY by 10 043, 047

The sign bits of x and x* are ignored and left clear except when a non-numeric character is used with the 043.

5. DIVISION 044, 045, 046

(a) If n = 0 the division is not performed but overflow is set.

x: is left undisturbed.

(b) The sign bit of x* is completely ignored.

(c) It is possible for rounding in the 045 instruction to cause overflow, or to recover overflow set by the division process.

(d) The remainder always satisfies the equation $r = x' = x - nx^{*'}$

6. SUBROUTINE ENTRY 070

It is the "programmers I.N.R." which is stored in X. i.e. in normal mode datum is subtracted from the contents of the "machine I.N.R.". before storage. (The I.N.R. consists of V, C, 7 mode bits plus 15 address bits). V and C are left clear but the mode bits are unchanged.

7. SUBROUTINE EXIT 072

N is datumed. The whole of the I.N.R. is restored but V is left set if already set. C IS CLEARED.

8. SHIFT INSTRUCTIONS 110, 111, 112, 113

The number of places shifted is equal to N_s if N_s is $\leq$ 127 or equal to N_s (mod. 64) + 65 if $N_s \geq 127$.

9. SINGLE LENGTH SPECIAL RIGHT SHIFT 112

Overflow is always left clear even if rounding "overflows". There is only one value of x for which this can arise. i.e. if V is set and X contains all ones the result of a single shift is -1.0 with V clear.

10. NORMALIZE 114, 115

(i) N = 0 or, N(mod. 4096) $\geq$ 2048

x: is set equal to zero and V is cleared.

(ii) Not case (i), V Clear

x: is shifted left (arithmetically) and the ten least significant bits of N are counted down once per shift until either:-

1. The sign bit of x: is different from the 2^{-1} bit.

or 2. N = 0 (ten least significant bits).

or 3. The six least significant bits of N become zero for the second time. i.e. x: must have been equal to zero.

In case 1 the nine least significant bits of the new N replace the nine least significant bits of x* and V is set if the tenth bit of N is not zero. In cases 2 and 3, x: is set equal to zero and V cannot be set.

(iii) Not case (i) V Set

First x: is shifted one place right (special shift). Then the nine least significant bits of N replace the nine least significant bits of x* and one

is added to x*. V will be cleared unless there is a carry from the ninth bit of x* into the tenth (from the least significant end) or unless the tenth bit of N is not zero.

NOTE

(a) In the case of the 114 instruction x* is first set equal to zero and the shifts are single length without rounding.

(b) In all cases the sign bit of x* is ignored. The result in x* is always positive except when there is a carry into the sign bit as a result of adding one to x* after a right shift. V will always be set if this happens.

11. NO OPERATION 123

This instruction does not clear the carry flip-flop.

W. R. Whittall