

FLOATING POINT DATA WORD FORMAT (STRETCH)

Scheme I

E. A. Voorhees and C. R. Blancett

Characteristics of Scheme I

1. Scheme is designed for normalized and unnormalized modes of operation.
2. Bit indicators in data words:
 - a. e (Epsilon) for underflow indication.
 - b. I for overflow indication.
 - c. Z (?) if needed to facilitate zero mantissa detection. (This write-up does not assume a Z bit.)
3. Exponent overflows and underflows can occur only when at least one of the operand mantissas is non-zero.
4. Let Q represent any floating point number with a zero mantissa. (It may be that Q should be further restricted.)
5. Let N represent any floating point number with a non-zero mantissa.
6. If I or e exists in either operand, the primary operation (+, -, x, ÷) is not performed. (Possibly normalization of the result of normalizing operations should be performed as in $N + Q_e$?) (Such normalization is assumed in this write-up.)
7. If one or both of the operands is Q, the primary operation is not performed. Again, it may be desirable to perform result normalization (if normalizing operation) as in $N + Q$. This is done in this write-up.
8. Four types of data words exist:
 - a. N: A number with a non-zero mantissa which is not the result of any exponent underflows or overflows.
 - b. Q: A word with a zero mantissa and which was not produced by exponent underflow.
 - c. N_I : The result of an operation which caused exponent overflow. (Note Floating Point Divide Table.)
 - d. Q_e : A word with a zero mantissa. When an operation causes an exponent underflow (" N_e ") the mantissa is set to zero and the e-bit inserted. For this reason, " N_e " does not exist in this scheme. Exponent clearing could also be done if desired. *
9. The main philosophy concerning N_I in this scheme is that the I in the word dominates and results from an " N_I " operand are generally N_I 's. Since the mantissa sign of an N_I may be useful information, this sign may be reversed in an operation of $Q - N_I$ or $Q_e - N_I$.

* Since no operation (as defined) can produce a " Q_I ", it does not exist. The same is true for " Q_{Ie} " and " N_{Ie} ".

Characteristics of Scheme I (Continued)

10. The mantissa sign for Q_e may be kept to indicate information regarding the formation of Q_e .
11. Operand (argument) mantissas are assumed in the following tables to be either normalized or unnormalized.

SOME SPECIFIC DEFINITIONS OF Q , Q_e , and N_I .

Notation: $[\text{---}] [\text{---}]$ means $[\text{Exponent}] [\text{Mantissa}]$.

A. Signed ExponentsSystem 1

$$Q : [-(2^{12} - 1)] [\pm 0] *$$

Q_e : Q with e indication.

N_I : Presence of I bit.

* $[\pm 0] [\pm 0]$ is artificial since it falls numerically "next" to $\pm \frac{1}{2}$.

System 2 (Better if normalized and unnormalized operations.)

$$Q : [\text{any exp.}] [\pm 0]$$

Q_e : Q with e indication.

N_I : Presence of I bit.

Note: N_I could be represented in the above two systems by all ones except for the $+$ exponent sign and no e bit ($-\infty$). The sign of the mantissa could be determined algebraically if possible.

B. Modular ExponentsSystem 1

$$Q : [\pm 0] [\pm 0]$$

Q_e : Q with e indication.

N_I : Presence of I bit.

System 2 (Better if normalized and unnormalized operations.)

$$Q : [\text{any exp.}] [\pm 0]$$

Q_e : Q with e indication.

N_I : Presence of I bit.

Note: N_I in the above two systems could be represented by all ones except for the e bit. The sign of the mantissa could be determined algebraically if possible.

SOME SPECIFIC DEFINITIONS OF Q , Q_e , and N_1 (Continued)

The preferred choice is probably System 2A or System 2B since it is anticipated that the machine will have unnormalized operations.

UNNORMALIZED OPERATIONS

- A. The usefulness of unnormalized operations seems to be dependent upon the division technique. The 704 technique for division seriously restricts the application of unnormalized operations. What seems to be needed is the elimination of Divide Check for division not involving a zero divisor. This might be obtained by temporarily normalizing the divisor before division. (This would even be desirable for normalized operations since unnormalized arguments are assumed to exist.)
- B. Exponents are assumed to be carried along with the Q 's.
- C. The details of unnormalized $+$, $-$, \times , \div could be as follows:

(All shifts involved are assumed to be associated with the proper adjustment in exponents.)

- Addition:
- (1) Match exponents of operand and operator by shifting right the one with the smaller exponent.
 - (2) Add fractions.
 - (3) Shift right one on overflow.

Multiplication:

- (1) Shift left the factor with the larger fraction until the factor is normalized.
- (2) Multiply fractions and form exponent.
- (3) Shift left one if the first bit of the product is zero.

Division:

- (1) If the dividend has the larger fraction shift right until the divisor fraction is the larger.
- (2) Shift the divisor left until normalized.
- (3) Divide the fractions and form exponent.
- (4) Shift left one if the first bit of the quotient is zero.

TRIGGERS AND BREAK-INS (Overflow - Underflow)

Assume four triggers:

Hi Ov	:	1		Lo Ov	:	3
Hi Un	:	2		Lo Un	:	4

Example: In the case of a division operation, exceptional events with regard to the quotient would have reference to Triggers 1 or 2; those of the remainder would have reference to Triggers 3 or 4.

Triggers are always turned on by exceptional events, i.e. exponent underflowing and/or overflowing.

There are two bits needed in operation (The usual condition of operation is underlined):

- 1 for Break-in or No Break-in. (Break-in, BI, is executed at end of operation.)
- 1 for Reset or No Reset. (Reset is executed at beginning of operation.)

When an exceptional event occurs, (a) the corresponding trigger (or two triggers) are turned on, and (b) a Test is made for Break-in or No Break-in.

If Break-in: Transfer to FM location 1,2,3, or 4, depending on trigger setting. If more than one trigger is on, transfer to the least number indicated. (The least number is the more important situation(?),) Do not reset trigger or triggers but do store location of Break-in instruction.

If No Break-in: Leave trigger(s) set, and do not set "Location Preserving Register."

Note: I and e are inserted as shown in tables.

The Reset bit resets all four triggers and clears the contents of the "Location Preserver" before the execution of the operation.

Note that a single instruction can rest before operation execution and Break-in after execution.

Also note that Break-in (dependent on triggers) does not have to be done during the operation which caused the trigger setting but may be done when desired. It is felt that this feature will be more useful than immediate Break-in in auto-coding since testing can be done at end of equation.

DIVIDE CHECK

One can assume a separate trigger: D.C. : 5
and an associated transfer to FM location 5.

A simpler scheme, if only division by zero is exceptional, would be to include the Divide Check case as a Hi Ov. (See Ov-Un Triggers and Break-ins.) and not introduce any additional lights, triggers, bits in operations, etc.

RANDOM OBSERVATIONS

1. If the machine can examine 48 zeroes as fast and as easily as with a Z bit, this technique is preferred since then hardware (and time?) would not be needed to test for 48 zeroes and insertion of Z bit.
2. Arithmetic Operations do not take place of OR of I's, e's, (Z's?) is 1.
3. Cancellation indication could be an additional bit (or 2) in the data word.

1 bit :

	C	
--	---	--

c=0 : Less than $\frac{1}{2}$ word cancellation.
 c=1 : More than $\frac{1}{2}$ word cancellation.

2 bits:

	C ₁	C ₂	
--	----------------	----------------	--

C₁ C₂

0	0	: Less than $\frac{1}{4}$ word cancellation.
0	1	: More than $\frac{1}{4}$, less than $\frac{1}{2}$.
1	0	: More than $\frac{1}{2}$, less than $\frac{3}{4}$.
1	1	: More than $\frac{3}{4}$.

(Would Break-in Techniques be desirable--more bits from instruction-?)


4. Should exponent "shortening" be done by external or internal switch, or by a more elaborate technique?
5. Should an internal switch be provided to halve "effective" mantissa lengths to provide unused bits to be used by the programmer in any way he desires and which are not disturbed or modified by arithmetic operations? (Would this be more flexible if done in quarters?)
6. Because of the nature of the definition of the BI bit in the operation, (i.e. do BI unless coder overrides it by "no BI") one may argue that an exponent underflow the definition is backwards in that the usual case is to ignore underflow. On the other hand, the definition is not backwards for exponent overflow. It may, therefore, be desirable to have two BI bits in the operation: one for underflow and the other for overflow.

NOTES REGARDING THE TABLE

1. Tables for both Normalized and Unnormalized Operations are shown. With either type of operation, the arguments (operands) are assumed to be unnormalized since normalized numbers can be thought of as a subset of unnormalized numbers.
2. Boxes which are identical to symmetric boxes are not shown.
3. Each box is divided into two parts, "BI" and "No BI". Results of operations (A and B registers) are shown depending on differing situations. The notation "CCEN" means that the given example cannot create an exceptional number, i.e. no exponent underflow or overflow can be developed during this operation. However, Break-in can occur due to the setting of a trigger(s) during a previous operation, in which case the A and B registers will probably have been modified by the present operation.
4. The purpose of the BI column is to demonstrate that, in many cases, useful information can be left in the A and B registers for subroutine or special code usage. If the instruction forbids Break-in, the entry in the "No BI"

NOTES REGARDING THE TABLE (Continued)

column represents the A and B registers. In all cases it is assumed that the original operands are available from the S register and the A' and B' registers.

5. The blackened rectangles, , mean that the primary operation (+, -, x, ÷) is not performed. (See Nos. 6,7, on p. 1)
6. Symbols in parentheses mean that the author cannot determine the more desirable choice. Symbols not in parentheses represent differing results.
7. In those cases where the A-register mantissa and the B-register mantissa are both zero, it is assumed that no normalization occurs. Otherwise, normalized +, -, x, and ÷ are assumed to normalize the non-zero result.
8. The result of an operation involving operands, both of which are Q's, is assumed to produce a Q whose exponent is computed by the usual rules for exponents.

NORM. +, -

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	N	-	N
see 1	Q	CCEN	N _I		see 4	Q _e	see 4	Q _e
see 2	Q _e							
see 3	N _I							

UNNORM. +, -

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	N	-	N
see 1	Q	CCEN	N _I		see 5	Q	see 5	(Q) or (Q _e)
see 3	N _I							

1. Due to total cancellation and treated here as CCEN
2. Due to partial cancellation and normalization, leave unnorm. result.

3. Due to mantissa overflow causing exponent overflow. Leave result since A-register is assumed to have an exp. spill bit.
4. Due to normalization only. Leave unnorm. result.
5. Due to exp. of Q. (CCEN)

NORM. X

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	Q	-	Q _e
see 6	N _I	CCEN	N _I		see 4	Q	see 7	(Q _e) or (Q)
see 7	Q _e							
see 4	Q _e							

UNNORM. X

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	Q	-	Q _e
see 6	N _I	CCEN	N _I		see 7	Q _e		(Q _e) or (Q)
see 7	Q _e							

6. Due to exp. calculation. Leave result described in 3.
7. Due to exp. calculation. Leave result described in 3.

NORM. ÷

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	Q	-	(Q _e) or (Q)
see 7	N _I	CCEN	N _I		see 8	Q	see 8	(Q _e) or (Q)
see 7	Q _e							
see 4	Q _e							

UNNORM. ÷

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N	-	N	-	N	-	Q	-	(Q _e) or (Q)
see 7	N _I	CCEN	N _I		see 8	Q	see 8	(Q _e) or (Q)
see 7	Q _e							

DIVISOR

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N _I	CCEN	(Q _e) or (N _I)	CCEN	N _I	CCEN	Q	CCEN	(Q _e) or (Q)
DC	NO DC		DC	NO DC	DC	NO DC	DC	NO DC
Q	see 8	N _I	see 8	N _I	see 8	Q	see 8	(Q _e) or (Q)

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
N _I	CCEN	(Q _e) or (N _I)	CCEN	N _I	CCEN	Q	CCEN	(Q _e) or (Q)
DC	NO DC		DC	NO DC	DC	NO DC	DC	NO DC
Q	see 8	N _I	see 8	N _I	see 8	Q	see 8	(Q _e) or (Q)

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
Q _e	see 8	N _I	see 8	N _I	see 8	(N _I) or (Q) or (Q _e)	see 8	Q _e

	N		N _I		Q		Q _e	
	BI	NO BI	BI	NO BI	BI	NO BI	BI	NO BI
Q _e	see 8	N _I	see 8	N _I	see 8	(N _I) or (Q) or (Q _e)	see 8	Q _e

8. What could be useful?
 (These boxes are not needed if SA on p. 3 is incorporated. It might then be desirable to replace "DC" by "BI")