Memo:

Recommendations concerning Noisy Mode

Memo: Calculational Experiment on noisy mode.

I. Intro

II description of interpretive Fl Pt.

III. description of noisy mode

IV. Results

Tables & graphs

~~is a good indication of the accuracy of the results~~

the resulting change in the final answers should be closely equivalent to that which one would achieve from doing the calculation with a computer which has a shorter word length for the mantissa.

II.

Although the basic idea of the noisy mode is appealingly simple, when one begins to look at it in detail one finds all of the old significance analysis difficulties still lurking behind the scene.

The approach in the present study is an experimental one. The argument is that if the noisy mode can be shown to work in a few real cases without displaying too many unpleasant properties, there is a good chance that it will work in many.

Project 17000 Dec 1, 1958

FILE MEMO

SUBJECT: Results of Some Calculational Experiments using Different Types of Noisy Mode Floating Point.

I. Introduction:

The problem of obtaining a meaningful measure of the significance in floating point calculations has troubled numerical analysts for many years. The difficulty inherent in such investigations has caused some well known workers in the field to discourage the use of floating point altogether.

Only Some of the new computers being built or planned now, such as the University of Chicago computer, contain special devices for keeping track of significance. The "Noisy Mode" in STRETCH is another method by which a relatively simple hardware can give indications of the accuracy of a specific problem.

The basic idea of the Noisy Mode is that a measure of the accuracy of a given problem is obtained by running it once with and once without "noise", the "noise" being a deliberate change introduced in a low order part of the mantissas of floating point operations.

~~[illegible] any change in the [illegible] answers of the problem should~~

Significance analysis is also disappointing in that a long analysis results in theoretical bounds for one particular problem type only. It usually says nothing about a specific case of the problem, and usually cannot be applied to another problem, even though similar in arithmetic.

A person who is interested in doing problems is much more concerned about the accuracy of his specific problem, than he is in general theorems. This has raised the desire for some built-in computer hardware which will somehow do error analysis on the specific problem being done.

Load: $(2^m M) \rightarrow (2^a A)$

1. Modify sign of M by S modifier, ~~place~~ place in R, place m in r.
2. If $M \equiv 0$, skip to 4.

3 Clear MQ, put R in acc, do same procedure as Add #9.

4. Place R + r in accumulator.

Store: $(2^a A) \rightarrow (2^m M)$

1. Modify sign of A by M, place in R, place a in r.
2. If $A \equiv 0$ skip to 4.
3. Do same procedure as Add #9.
4. Place R + r in m + M, leave a and A in acc.

Multiply: $(2^a A) \cdot (2^m M)$

1. Modify sign of M by S modifier,
2. add $a + m = r$.
3. Multiply A by M in fixed point (double product in acc-MQ)
4. (Case I noise inserted here.)
5. Test if result is zero, if so skip to 7.
6. Do same procedure as Add #9
7. Place r and R in acc., low order part of product in MQ.

Divide: $(2^a A)/(2^m M)$

1. modify sign of M by S modifier,
2. Test if $M \equiv 0$, transfer to divide error subroutine.

A brief description of the floating point operations follows:

Add: $(2^a A) + (2^m M) = (2^r R)$

1. Modify sign of M by S modifier
2. Compare a and m, place mantissa of smaller in acc., larger in R, larger exponent in r.
3. If $|a-m| > 27$ no addition required, skip to 10.
4. Shift acc. right $|a-m|$ places.
5. Add R to double length acc + MQ in fixed point. (watch MQ sign)
6. Test if result $\equiv 0$, if so set R + r to zero & skip to 10.
7. (Case I noise inserted here.)
8. Test for overflow in addition, if none skip to 9.
   (a) If N and overflow, shift R right 1 and increase r by 1.
   (b) If U and overflow, transfer to overflow error subroutine.
9. (a) If N, test for leading zeros in result mantissa.
   (1) If there are zeros:
   (Case II noise inserted here.)
   Shift acc-MQ left by number of zeros, reduce exponent r by amount of shift.
   (2) If there are no leading zeros, leave result as is.
   (b) If U, leave result as is.
10. Place R and r in accumulator.

IV. The Interpretive Floating Point Program.

In order to perform the noisy mode experiments it was first necessary to code an interpretive floating point program for the 704. The built-in floating point hardware could not be used because one needs access to the numbers at intermediate steps in the execution of the operations.

The floating point code uses the same word format as the regular 704 floating point format, but its sign modifiers and normalized-unnormalized modifiers are patterned after STRETCH. It also uses STRETCH philosophy in retaining double precision intermediate results and handling exception cases. No double precision operations were included.

Using the operation definition pseudo-op (OPD) in SAP, the following 20 pseudo ops were defined:

using O = opposite sign N = normalized
S = same sign U = unnormalized

Basic ops.	Floating point pseudo-ops defined			
add	AON	AOU	ASN	ASU
mpy	MON	MOU	MSN	MSU
div	DON	DOU	DSN	DSU
load	LON	LOU	LSN	LSU
store	SON	SOU	SSN	SSU

3. Subtract $a - m = r$.

4. If $A = 0$, set R to zero and skip to 9.

5. If $|A| - |M| \geqq 0$ shift acc + MQ right until $|A| < |M|$. Increase r by one for each place shifted.

6. Divide by M in fixed point.

7. (Case I routine inserted here).

8. Place quotient in acc, clear MQ, do same procedure as Add #7.

9. Place $R + r$ in acc.