

Eastern Illinois University

The Keep

Armstead Publications and Presentations

Betty Jo Armstead

1969

Tagged Arithmetic

Paula J. Bettinger

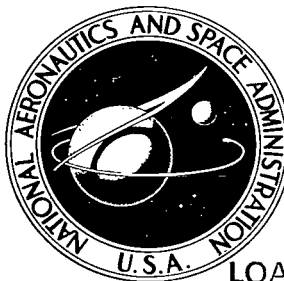
Andrew M. Manos

Betty Jo Armstead

Follow this and additional works at: https://thekeep.eiu.edu/archives_armstead_publications



Part of the [Physical Sciences and Mathematics Commons](#)



NASA TECHNICAL NOTE

NASA TN D-5370

c. 1

NASA TN D-5370

LOAN COPY: RETURN
AFWL (WLIL-2)
KIRTLAND AFB, N M



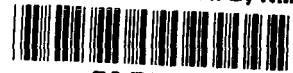
TAGGED ARITHMETIC

*by Paula J. Bettinger, Andrew M. Manos,
and Betty Jo Armstead*

*Lewis Research Center
Cleveland, Ohio*



TECH LIBRARY KAFB, NM



0132292

TAGGED ARITHMETIC

By Paula J. Bettinger, Andrew M. Manos, and Betty Jo Armstead

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 - CFSTI price \$3.00

ABSTRACT

A special tagged arithmetic system has been developed for use with FORTRAN programs written to process experimental data. The tagged arithmetic system carries a condition code with every numerical value and uses a special output to call attention to answers computed by using questionable input data. The questionable input data may result from instrumentation or data recording system malfunctions, which can cause ill-conditioned calculations that result in process-time faults or error conditions.

TAGGED ARITHMETIC

by Paula J. Bettinger, Andrew M. Manos, and Betty Jo Armstead

Lewis Research Center

SUMMARY

A special tagged arithmetic system has been developed for use with FORTRAN programs written to process experimental data. The tagged arithmetic system carries a condition code with every numerical value and uses a letter code on the output to call attention to answers computed by using questionable input data. This system eliminates stops from process-time faults or error conditions and permits calculations to be completed with questionable data. It reduces the programing effort formerly required to detect faulty input and thus avoid ill-conditioned calculations. Reprocessing of data is significantly reduced, and the user's confidence in results is increased.

INTRODUCTION

Most of the experimental data processed at Lewis are digitized quantities of physical measurements recorded by central and remote data recording systems. Since the data are a result of direct measurements, some of the data recorded might lead to impossible operations (e. g., division by zero or the square root of a negative number). For the most part, these data are due to instrument failures or to test point settings where a small error in data causes the calculations to be ill-defined. In many cases, such data can produce plausible, but incorrect, results. Complete error analysis of the source of these faults is impractical to perform internally in the computer while processing the data. Therefore, the calculations must be programed to proceed in the best manner possible, and the questionable data flagged for analysis later. A tagged arithmetic system was designed to aid in the analysis of the experimental data and subsequent calculations performed with the data.

The system was first developed by Mr. L. Richard Turner of Lewis. It was implemented in a special interpretive data processing system in use on the UNIVAC 1103 com-

puter. The present tagged arithmetic system was written for use with FORTRAN IV programs running on the IBM 7094II/704X direct couple systems. It has been in use since the spring of 1966. The system has proved to be extremely useful and thus was also implemented for FORTRAN programs running on the 360/67 with Operating System 360. We know of no other software system or computer designed to carry a condition code along with the data throughout the calculations.

OVERVIEW OF SYSTEM

The tagged arithmetic system allows processing of data to proceed regardless of the quality of the data. Questionable input data and further calculations based on it are tagged as the data are processed. The appearance of a letter printed immediately to the right of the results on the output alerts the research engineer that his results may be questionable. This letter enables him to analyze the causes of the flag and to make judgment as to the validity of the results. Figure 1 shows sample output from the tagged arithmetic system.

The basis for tagging questionable data consists of one or more of the following elements:

- (1) Instrument failure
- (2) Digitizer limitations (data exceeding range)
- (3) Faulty data syntax
- (4) Parity errors
- (5) Violation of calibration limits
- (6) Programed validity test failure
- (7) Impossible arithmetic operation

A word can be tagged at recording time either by the engineer's choice or by the recording hardware (i. e. , parity errors). It can be tagged at execution time either by the programmer's choice or by the operating system as a result of an impossible arithmetic operation or an operation with a previously tagged word.

The four basic categories of tags are defined as follows

(1) Input tag or engineer's code-out: An input tag must be generated by the data recording systems and/or subprograms which analyze the digitized input of experimental data. This tag is inserted for data which exceed the limit of a digitizer, have parity errors, or have faulty syntax. An engineer can tag data caused by faulty instrumentation by use of the central data recording systems or remotely by use of a plugboard at the site or through communication with the processing programs. Any calculations subsequently based on data with an input tag would also contain the input tag.

(2) Programed validity test failure: This tag is generated by the programmer and will take on whatever meaning he desires. It could be used to specify that the number of iterations has exceeded a limit or that a data point has an excessive deviation from a curve fit.

(3) Computer overflow or impossible operation: This tag is reserved for use by the floating-point trap error package which resides in the system. No other source is permitted to insert this tag. The error package (NASA Lewis Research Center 7094II/7044 Direct Couple Monitor Program Reference Manual) in use on the Lewis direct couple computer can establish the existence of the tagged arithmetic system within a particular program and insert tags whenever an impossible operation occurs. For example, the tag might be caused by dividing by zero or taking a square root or logarithm of a negative number.

(4) Calibrations outside a declared range: In processing experimental data an engineer must specify calibrations for the data. A tag is generated whenever a calibrated value falls outside the range for which the calibration has been specified as valid.

The value of tagged arithmetic depends on its proper use by the engineer and programmer. When proper precautions have been taken at the recording facility and in the program to insert the necessary tags, data faults will naturally propagate throughout the processing of the data, enabling the engineer to analyze the results of his experiment. The appearance or absence of tags on the printed results gives the engineer an easy clue to the reliability of the processed data.

Tagged arithmetic can be used in other applications. With the exception of the impossible operation, the user can give his own meaning to the categories of tags and use them to trace the effects of specific data or the flow within a program.

LIMITATIONS

The tagged arithmetic system was designed with the programmer, as well as the research engineer, in mind. It was considered undesirable to place restrictions on the programmer beyond those that were already part of the operating system; however, a few minor restrictions were found necessary.

Programing Restrictions

The following restrictions are placed on programs using tagged arithmetic:

(1) Library routines ATAN, ATAN2, GAMMA, ALGAMA, ERF, TANH, TAN, and COTAN must not be referenced for any data that might be tagged. Instead, the entry

names TATAN, TATAN2, TGAMMA, TLGAMA, TERF, TTANH, TTAN, and TCOTAN are provided to handle tagged arguments. All other library routines may be referenced normally.

(2) On large jobs that require overlay, non-FORTRAN library subprograms must be included in the main link.

(3) Output fields for numbers that may be tagged must be large enough to accommodate the alphameric character that represents the tag. In the field specifications Ew.d, Fw.d, and Gw.d both w and d are reduced by 1 to write out a tagged number. In the programmer's format specification, d must not be zero.

(4) The load-time debugging package provided by IBM (IBJOB Processor Debugging Package (\$IBDBL, *DEBUG, etc.)) may be used if two rules are followed:

(a) Debug requests may only be made at the beginning of a FORTRAN statement. (This is normal for FORTRAN programs.)

(b) Debug requests may not be made on any statement that may result in a subroutine CALL as its first operation (i.e., CALL, READ, WRITE, $X = \text{SQRT}(X)$, etc.). Requests may not be made on a CONTINUE statement which immediately precedes the above types of statement.

Tagged numbers printed by debug requests will appear half the magnitude they are and will not be followed by an alphabetic tag since the debug package does not make any special provision to recognize tagged numbers.

(5) The standard method of handling arithmetic errors (NASA Lewis Research Center 7094II/7044 Direct Couple Monitor Program Reference Manual) is slightly altered when using tagged arithmetic. The differences are as follows:

(a) The result of all illegal arithmetic operations, except underflows, will be tagged with a computer overflow or impossible operation tag.

(b) Division by zero will always be treated as an overflow (i.e., the result is set to the largest representable floating-point number). Otherwise, the programmer's control of errors is not altered.

(6) FORTRAN logical expressions do not make any provision to handle tagged numbers. An arithmetic expression or variable which is tagged is used in comparisons as follows:

(a) A tagged zero is not equal to zero.

(b) A tagged nonzero is half magnitude.

Storage Requirements and Timing

Routines of the tagged arithmetic system add less than 1000 words to the storage required for the basic programs. Inclusion of all the optional routines adds less than another 500 words.

A typical data processing job may require twice as much execution time with the tagged arithmetic system. The disadvantage of a longer run time is more than overcome by the advantages of being able to complete a run in spite of questionable data and to evaluate the results with any errors automatically propagated to the resulting output. This reduces the necessity for reprocessing because of questionable data.

IMPLEMENTATION

The tagged arithmetic system is implemented for use with programs written in FORTRAN IV. Several modifications were made to the operating system to accommodate tagged arithmetic. To further facilitate the scanning and alteration of programs, the system library was ordered to separate routines which need to be scanned from those which do not. These changes in no way affect execution of jobs which do not use the tagged arithmetic system.

Tagged Format

The sign and characteristic portions of tagged and untagged data words are the same. However, a tagged data word in machine memory is always unnormalized by shifting the mantissa portion one bit position to the right. This gives the tagged data word a unique form.

The four low-order bits of a tagged data word are used to express four categories of faults, as shown in appendix A. Each bit position in the four-bit tag represents a different category of fault. The tag is then the logical OR of the faults which appeared in operands that produced this data word. There are 22 bits to express the significant mantissa, which is considered sufficient for questionable data. Untagged data words suffer no loss of precision. The tagged arithmetic system uses only single-precision (36 bit) floating-point words.

Instruction Alteration and Recognition

The tagged arithmetic system inserts a mask into every floating-point operation (addition, subtraction, multiplication, and division) and calls to certain mathematical functions. The first octal digit of each of these commands is 0 and is changed to 5. When this changed command is executed, the location of the operation is stored, and control is transferred to a fixed location in the core (i. e. , a STORE AND TRAP occurs).

This fixed location then sends control to the arithmetic portion of the tagged arithmetic system. The system removes the mask to recover the original floating-point operation. Control is thus transferred to the system before each floating-point operation to check for the existence of a tagged operand. If there is no tagged operand, the operation is performed normally (after removing the mask). Otherwise, the special tagged arithmetic operations must be used to preserve and propagate the tags, as well as to perform the arithmetic operation specified.

Tagged Arithmetic Subprograms

The tagged arithmetic system includes 13 subprograms. A description of each of these subprograms is given in appendix B. One subprogram is available to the programmer for invoking the tagged arithmetic system. Eight subprograms are available to the programmer for examining the tags of numbers, for inserting tags, and for special entries to library routines that would not otherwise be able to handle the tagged numbers. The remaining subprograms are called automatically and do not need to be known by the programmer.

The programmer must call the subprogram TAGSCN to activate the tagged arithmetic system before executing any other instructions. This subprogram determines the areas of core storage to be scanned for floating-point arithmetic operations. Two areas are scanned, the object program loaded by the programmer and all standard library subprograms. TAGSCN calls the SCANTG subprogram, which scans the necessary portion of the core and replaces floating-point operations and certain library subprogram calls so that they will be tested for tagged arithmetic when executed. The replaced operations are addition, subtraction, multiplication, and division, or any of the transfers to the functions SIN, COS, EXP, SQRT, ASQRT, ALOG, and ALOG10.

During execution, the TAGOPE subprogram is entered when an operand is a tagged number. Floating-point arithmetic operations and transfers to the functions SIN, COS, EXP, SQRT, ASQRT, ALOG, and ALOG10 are examined by this subprogram. The tag or tags of the operands are removed, the indicated operation is performed, and the result is flagged with the combined tags of the operands. The subprogram TAGPF is called for the special library routines TATAN, TATAN2, TGAMMA, TLGAMA, and TERF written to replace routines that operate on the characteristic and mantissa of a floating-point argument separately rather than using the standard machine floating-point operations.

Operating System Modifications

Several modifications were made to the operating system to accommodate the tagged

arithmetic system. These in no way affect the execution of jobs which do not use tagged arithmetic. The modifications were as follows:

(1) Reference points were inserted so that information available to the operating system could also be referenced by the tagged arithmetic system.

(2) The standard error package in use at Lewis reports arithmetic errors that cause a machine overflow, underflow, division by zero, or an attempt to use an illegal argument to a standard library arithmetic function. The standard error package was modified so that it would insert the appropriate tag into the result of an impossible operation. Error reports are not given if an operand was previously tagged.

(3) Output conversion of floating-point numbers was modified to recognize a tagged number. The conversion normalizes the number and adjusts the field specification to include an alphameric character to represent the appropriate tag.

(4) The routine which overlays links or large jobs was modified to call for a scan of the appropriate area each time a new link is read into the memory. A table of subroutine locations and link numbers is preserved in lower memory by the loader. The loader has been modified to save this table for the Lewis error package.

(5) To further facilitate the scanning and alteration of programs, the system library was reordered to separate routines which need to be scanned from those which do not.

CONCLUDING REMARKS

A special tagged arithmetic system has been developed at Lewis Research Center for use with FORTRAN programs written to process experimental data. The tagged arithmetic system carries a condition code with every numerical value and uses a letter code on the output to call attention to answers computed using questionable input data.

The programing restrictions for use of the tagged arithmetic system with FORTRAN programs are relatively minor. Considerable programing time is saved by use of the system. The system performs decisions and flags results which otherwise would have to be programed for each experimental test. The disadvantages of the longer run time are more than overcome by the advantages of being able to complete a run in spite of questionable data and to evaluate the results with flags automatically propagated to the resulting output. This reduces the necessity for reprocessing because of questionable data.

The tagged arithmetic system has applications other than processing data which are the results of physical experiments. The bit structure which includes a tag and propagates it through calculations is useful in tracing and general program debugging.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, June 9, 1969,
129-04-06-03-22.

APPENDIX A

BIT STRUCTURES OF TAGGED REAL NUMBERS USED IN FORTRAN

PROGRAMS WITH FLOATING ARITHMETIC TRAPS

Let S, E, N, and F represent the sign, exponent, numeric, and fault bits of the real number, respectively. The effect of tagging on the zero and nonzero bit structures can be represented as follows:

S 00000000 000000000000000000000000 0 0 0 0	Untagged zero data
S 00000001 000000000000000000000000 $F_8 F_4 F_2 F_1$	Tagged zero data
S EEEEEEEE 1NNNNNNNNNNNNNNNNNNNNNNNNNN N N N N	Untagged nonzero data
S EEEEEEEE 01NNNNNNNNNNNNNNNNNNNNNNNNNN $F_8 F_4 F_2 F_1$	Tagged nonzero data

The interpretation of the fault bits is as follows:

F_8	Input tag or engineer's code-out
F_4	Failure of programmed test for validity
F_2	Computer overflow or impossible operation
F_1	Data word outside declared range

Example:

20140000000	Octal representation of a floating-point untagged 1
201200000005	Floating-point 1 tagged with F_1 and F_4 bits

Tagged data output can warn the engineer that a data word is doubtful or faulty. The output of a FORTRAN program using F, E, or G format conversion for real single-precision tagged data words will yield a literal tag appended to the right of the printed result. An example of this output using 016, F16.6, E16.6, and G16.6 on a normal zero, a zero with an input tag, a normal 1, and a 1 with both an input and an out-of-range tag is illustrated as follows:

Conversion	Untagged zero	Tagged zero	Untagged 1	Tagged 1
O16 (octal)	000000000000	001000000010	201400000000	201200000011
F16.6	0	0.00000H	1.000000	1.00000S
E16.6	0	0 00000E-38H	0.100000E 01	0.10000E 01S
G16.6	0	0.00000E-38H	1.000000	1.00000 S

The tag letters that can appear in the output, such as H and S in the previous illustration, can be interpreted by means of the following key:

Tag value	Tag meaning	Tag letter														
		A	B	C	D	E	F	G	H	S	T	U	V	W	X	Y
8	Input tag								*	*	*	*	*	*	*	*
4	Failure of programmed test for validity				*	*	*	*				*	*	*	*	*
2	Impossible operation		*	*			*	*			*	*			*	*
1	Out of declared range	*		*			*	*	*		*	*	*	*	*	*

APPENDIX B

SUBROUTINES USED WITH TAGGED ARITHMETIC

The subroutines used with tagged arithmetic are given in the following table:

Subroutine	Usage	Call sequence	Error message	Error condition	Optional exit control
NTAG	Finds tag of tagged number, if any	y = NTAG(x)	None	None	None
TAG	Tags a number	CALL TAG(x, n)	* ATTEMPT TO TAG A NONREAL GAVE A TAGGED ZERO	n is not real	\$LIBSAR
TAGSCN	Sets all floating-point operations for possible tagged arithmetic; determines proper error conditions	CALL TAGSCN	None	None	None
UNTAG	Removes tag from tagged number	y = UNTAG(x)	None	None	None
TATAN	Calls ATAN for argument that may be tagged	y = TATAN(x)	None	None	None
TATAN2	Calls ATAN2 for argument that may be tagged	y = TATAN2(x, z)	* ILLEGAL ARG. TAGGED ATAN2	x or z is non-real, nontagged	\$LIBSAR
TERF	Calls ERF for argument that may be tagged	y = TERF(x)	None	None	None
TGAMMA	Calls GAMMA for argument that may be tagged	y = TGAMMA(x)	None	None	None
TLGAMA	Calls ALGAMA for argument that may be tagged	y = TLGAMA(x)	None	None	None
^a SCANTG	Changes floating-point operations to tagged operations	None	None	None	None
^a TAGTRP	Tests operands for tagged numbers and performs normal arithmetic operations; sets up output for tagged numbers	None	None	None	None
^a TAGOPE	Performs tagged arithmetic	None	None	None	None
^a TAGPF	Provides linkage to function subprograms for tagged arguments	None	None	None	None

^aUsed automatically and are not called by the programmer.

Subroutine TAGSCN (Written in MAP)

This subroutine must be called by the program before any other statements are executed in order to use any tagged arithmetic during execution of the program. It determines the areas of core storage to be scanned for possible floating-point arithmetic operations. Two areas will be scanned, the object program loaded by the programmer and all standard library subroutines. Three areas are not scanned. These include the resident portion of the IBJOB monitor system itself; nonstandard, non-FORTRAN library routines; and that portion of core storage beyond the loaded object program and library. TAGSCN calls subroutine SCANTG.

Subroutine SCANTG (Written in FORTRAN IV)

This subroutine actually scans the necessary part of the core and replaces floating-point operations and certain library routine calls so that they may be tested for tagged arithmetic when executed. The first octal digit is set to 5, which will cause (1) the instruction counter to be stored and (2) an illegal operation trap. These operations are addition, subtraction, multiplication, and division, or any of the transfers to the functions SIN, COS, EXP, SQRT, ASQRT, ALOG, and ALOG10.

Subroutine TAGOPE (Written in FORTRAN IV)

TAGOPE is entered when an operand is a tagged number. The tag or tags of the operands are removed, the indicated operation is performed, and the result is tagged with the combined tags of the operands. Permissible operations are addition, subtraction, multiplication, and division, or one of the functions SIN, COS, EXP, ALOG, SQRT, ASQRT, and ALOG10.

Subroutine TAGPF (Written in FORTRAN IV)

This logical function will determine if the operand is a tagged number. If it is tagged, a function subprogram will be called to operate on the argument and the function will be set .TRUE.. This function is used by the special library routines that were written to replace routines that operate on the characteristic and mantissa of a floating-point argument separately rather than by using the standard machine floating-point operations.

Subroutine TAGTRP (Written in MAP)

This subroutine recognizes a command that may be a possible tagged arithmetic operation. If neither operand is tagged, the operation is performed normally. If either one is tagged, TAGOPE is called to perform the operation. This routine also controls the output conversion of tagged numbers in the E or F conversion. If a number to be written out is tagged, the alphabetic tag will appear in the rightmost position of the specified field, thus causing one less significant digit to appear.

Subroutine NTAG (Written in FORTRAN IV)

This function subprogram is used in a FORTRAN tagged arithmetic data processing job to determine whether a single-precision floating-point number is tagged. An integer is generated, the value of which depends on the fault conditions that apply to the number. The value of that integer is obtained by a reference to NTAG(X). The relations among the value of the integer, the letter used to tag this number in output, and these fault conditions are illustrated as follows:

Fault conditions that apply to number X	Tag bits of X	Value of NTAG(X)														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		Tag letter														
		A	B	C	D	E	F	G	H	S	T	U	V	W	X	Y
Input tag or engineer's code-out	8							*	*	*	*	*	*	*	*	
Failure of programmed test for validity	4			*	*	*	*				*	*	*	*		
Computer overflow or impossible operation	2		*	*		*	*		*	*			*	*		
Data word outside of declared range	1	*	*	*	*	*	*	*	*	*	*	*	*	*		

When X is untagged, NTAG(X) is zero. When X is not a single-precision floating-point number, such as when X is an integer, NTAG(X) is set to the value -31.

Subroutine TAG (Written in FORTRAN IV)

This subroutine is used to set specified fault bits, called the tag, of a real, single-precision, floating-point data word. The tag is used to specify fault conditions that apply

to the data word; the tag NN is defined as follows:

- NN=8 input tag or engineer's code-out
- NN=4 failure of programmed test for validity
- NN=2 computer overflow or impossible operation
- NN=1 data word has fallen outside declared range

Tags propagate through tagged arithmetic to the results of the calculation. When E, F, or G format conversions are used for output, the tags are reported as literal tags. Insertion of a tag is made by the statement

CALL TAG(X, N)

where

X data word to be tagged

N fixed-point variable or constant, sum of all values of NN to be inserted with this CALL

Define n congruent to N modulo 16.

- (1) When $n = 0$, **X** is left unchanged.
- (2) When **X** is untagged, the resulting tag of **X** is n .
- (3) When **X** is tagged, the resulting tag of **X** is the logical sum of n and the original tag of **X**.
- (4) An NN=2 in n is ignored. The programmer cannot control this bit. The relation among n , the letter tag in the output, and the fault bits is given as follows:

Tag bit	Value of n														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Tag letter														
	A	B	C	D	E	F	G	H	S	T	U	V	W	X	Y
NN=8									*	*	*	*	*	*	*
NN=4				*	*	*	*					*	*	*	*
NN=2		*	*			*	*			*	*			*	*
NN=1	*		*		*		*		*		*		*		*

When **X** is nonreal, such as an integer, the following error message is returned by the Execution Error Monitor:

* ATTEMPT TO TAG A NONREAL GAVE A TAGGED ZERO

Subroutine UNTAG (Written in FORTRAN IV)

This subroutine may be used to remove the tag from a number. If the number has been tagged, the accuracy is limited to the 22 bits available in the mantissa. The statement

$$Y = \text{UNTAG}(X)$$

stores the untagged value of X in Y . Programers are cautioned about the use of this routine so that the purpose of tagged arithmetic is not wasted.

Subroutines TATAN, TATAN2, TGAMMA, TLGAMA, and TERF (Written in FORTRAN IV)

These function names are used to reference the respective functions whenever there is a possibility that an argument may be a tagged number. These names should always be used by a program that uses tagged arithmetic.

Use	Do not use
$z = \text{TATAN}(x)$	$z = \text{ATAN}(x)$
$z = \text{TATAN2}(x, y)$	$z = \text{ATAN2}(x, y)$
$z = \text{TGAMMA}(x)$	$z = \text{GAMMA}(x)$
$z = \text{TLGAMA}(x)$	$z = \text{ALGAMA}(x)$
$z = \text{TERF}(x)$	$z = \text{ERF}(x)$

Each of these routines calls TAGPF. An entry into the normal library routine is then effected with an untagged argument. If any tag is present, it will be propagated in the function value before return to the user's program.

Figure 1

NASA-LEWIS RESEARCH CENTER CLEVELAND, OHIO PRELIMINARY DATA 01/15/69 CADDE-II RECORDED 1 3 21 50 40 438
 FACILITY PSL-2 PROG 0007 RDG 0137

BAROM = 14.36PSIA

AVD	1	2	3	4	5	6	7	8	9	10
	1-51.0232	-53.8371	-683.37	A-672.343	-3.44234	498.486	9399.21	9388.88	9393.47	12707.1
	11 12699.8	12699.8	20.4088	25.4037	18.7015	9.15292	4.52307	4.44765	4.56615	0.13301E-01
	21 9.08191	2.99680	-0.88719E-02	0	519.871	467.145	496.580	3.4254	H 2.24122	471.919
	31 422.152	0.20002	2.28923	83.5560	2173.47	2128.34	2056.98	2055.49	2105.87	2092.23
	41 440.781	437.771	442.083	438.825	440.029	437.821	442.083	438.825	438.976	438.875
	51 439.879	439.126	531.020	0.50005E-02	0.20002E-02	584.260	585.926	581.949	578.989	578.817
	61 586.395	585.029	581.306	577.613	577.398	580.706	578.602	576.709	574.900	578.688
	71 575.719	574.166	573.001	578.000	582.891	104.21	H 576.537	575.633	610.68	H 579.805
	81 582.334	948.56	H 577.484	579.118	582.934	574.037	574.900	609.68	H 579.976	528.38 H
	91 1015.22	1017.49	1016.35	1021.95	1027.55	1019.07	1017.49	1013.90	1020.20	1027.90 H
	101 1013.55	1011.63	1017.84	1025.36	1034.19	1014.34	1009.53	1011.37	1020.55	1028.60
	111 746.17	H 1918.59	-6042.2	S 1856.84	1765.48	2020.08	2147.08	2133.69	1931.8	H 611.32 H
	121 1836.38	1984.68	2083.97	2046.46	1851.61	1912.87	1928.27	1984.68	1930.48	1959.85
	131 1755.59	1765.91	611.32	H 1748.08	1528.38	1331.86	1565.04	1579.67	1502.97	1462.95
	141 1522.51	611.11	H 1575.49	1548.93	1458.10	1649.44	1712.37	1765.26	1738.23	1519.57
	151 1608.26	1707.04	1778.60	1646.95	1384.85	586.357	580.671	581.914	582.814	594.870
	161 590.053	572.17	H 587.251	586.144	581.443	578.393	576.627	851.070	504.362	508.767
	171 508.814	507.410	504.221	577.101	573.428	578.695	586.229	577.532	578.910	585.205
	181 599.906	599.822	600.575	624.006	602.495	602.745	634.594	670.481	779.850	755.995
	191 739.696	790.093	611.32	H 713.082	231.09	H 743.270	611.32	H 711.067	739.696	745.057
	201 677.890	674.972	611.32	H 732.544	726.955	750.415	779.182	719.125	769.377	611.32 H
	211 728.520	724.718	-15886.	S 753.094	734.556	611.32	H 738.578	708.604	727.178	597.179
	221 708.156	735.673	692.70	H 692.247	703.677	721.810	710.396	765.141	611.32	H 719.125
	231 709.500	-5450.8	S 763.357	611.32	H 747.290	718.230	757.334	704.797	709.500	712.635
	241 611.32	H 166.277	14.6525	479.663	0.30003E-02	507.083	0.40004E-02	12.7253	27.2927	42.9193

Figure 1. - Sample output from tagged arithmetic system.

DAMPR

1	0.97992	0.97992	0.97244	0.97992	0.98365	35.7006	35.7044	35.6968	35.7044	35.7006
11	14.2986	14.2907	14.3026	14.2986	14.3026	0.97917	0.97009	0.98825	0.97917	0.97917
21	87.7702	87.8156	87.7793	87.7793	87.7793	14.2929	14.2835	14.2835	14.2740	14.2335
31	0.97300	1.00384	0.97300	0.97300	0.97300	139.856	139.886	139.871	139.856	139.871
41	14.4107	14.3788	14.3629	-0.3374	A 14.3788	4.85059	4.83111	4.85059	4.85059	4.83890
51	4.84670	4.36461	4.36073	4.36849	4.37236	4.84280	4.84670	4.81942	4.80383	4.82331
61	4.83500	4.84280	4.81942	4.79604	4.85059	4.85449	4.82721	4.91298	4.79994	4.84280
71	4.81552	12.247	H 11.689	H 11.578	H 11.753	H 10.803	H 4.81552	4.7649	H 4.84280	4.80393
81	4.82331	4.79994	7.1899	H 11.246	H 11.246	H 7.2179	H 11.286	H 4.78825	4.83500	4.82331
91	4.81162	4.83500	4.81942	4.85059	4.85059	4.2600	H 4.2445	H 4.2638	H 4.2948	H 11.238
101	4.81942	4.83890	4.83111	4.23672	4.28708	4.28708	4.28321	4.28321	4.26771	3.2045
111	3.2314	H 3.2352	H 3.2237	H 4.60923	4.66757	4.72205	4.75709	4.79214	4.83111	4.83500
121	4.56259	4.65590	4.69481	4.72984	4.77656	4.79214	4.80383	4.22123	4.22123	4.22998
131	4.19800	4.21736	10.7388	11.0008	11.0664	11.0196	11.0570	11.141	H 10.9540	10.7763
141	11.0102	11.1132	10.066	H 11.2162	11.5067	11.0102	10.8511	11.4504	11.4129	11.3942
151	11.3099	11.3286	11.4879	10.5986	11.3099	11.1600	11.3192	11.2256	6.7686	H 11.5536
161	10.7295	10.7482	10.8230	10.4210	10.1221	10.8511	10.9072	11.2630	11.0196	9.84211
171	10.9915	11.1319	11.1694	10.7201	9.93541	10.8230	10.9540	13.887	H 11.0289	10.0101
181	10.9260	10.9634	10.9634	10.7108	9.82345	9.42259	8.71527	9.33875	8.62231	9.18044
191	8.58513	11.1881	11.1694	11.1225	8.69667	8.73386	8.52009	7.88886	8.03728	5.99976
201	6.99976	7.55517	10.8792	8.93847	9.52509	8.42718	8.38074	8.43647	8.64090	8.91055
211	9.02221	9.09666	8.46434	8.66879	8.13935	7.83322	8.03728	8.17648	8.38074	61.3882
221	3.2700	H 60.6570	61.5438	61.4971	63.0053	61.0771	60.8748	61.7926	60.6414	61.9537
231	62.8343	61.7149	61.9326	61.5438	62.2592	61.1082	61.5749	61.2794	61.1082	58.7107
241	14.155	H 57.0743	57.1522	13.645	H 55.2647	23.7039	23.6246	23.5928	23.7357	23.4499
251	9.52509	9.53441	9.77682	9.87942	9.55305	10.1501	10.6921	10.9353	10.6360	10.2155
261	10.1688	10.0007	10.2622	10.1314	10.5144	3.1472	H 10.3182	10.0194	10.8230	10.7014
271	10.1127	9.37601	9.32944	9.39464	8.83615	3.73271	9.21769	9.19906	9.24562	9.18044
281	9.42259	9.14321	9.09666	9.41328	9.46918	8.35288	8.78035	8.19504	8.32502	3.21634
291	3.21987	3.22753	3.22753	3.19306	8.89281	8.96997	8.88469	8.81977	8.94559	8.85223
301	8.79949	8.81977	8.84817	8.84817	3.12032	3.24669	3.22753	3.21987	3.21987	3.19306
311	3.20454	3.20454	3.21604	3.21987	3.20454	3.21221	3.22370	8.05583	7.66636	7.66536
321	7.50885	7.28660	7.15703	11.3286	11.6848	11.0570	3.1655	H 11.0570	10.7482	25.1004
331	50.0755	25.4969	28.1910	57.6823	58.7419	60.5481	36.6419	48.5410	30.0746	14.6819
341	14.3629	8.18576	3.23902	3.20071	-0.6268	A-0.6268	A 10971	10951	10898	11031

Figure 1. - Continued.

AVERAGES

PP	2	4.84085	TP	12	439.589	PT1	4	4.84670	PS1	4	4.36655	PT2	32	4.82879
PS2	6	4.27417	PT2A	3	4.83630	PT2B	5	4.82487	PT2C	5	4.84826	PT2D	3	4.81942
PT2E	-5	11.614	H PT2F	2	4.82916	PT2G	3	4.80903	PT2H	-5	9.6374	H PT2I	5	4.81864
PT2J	3	4.84020	PT2K	-5	5.6603	H PT2L	3	4.82981	PTC21	9	4.80989	PTC22	3	4.83630
PTC23	8	4.83939	PTC25	9	4.82678	PTC26	3	4.85580	PT23	25	11.1548	T23	10	581.768
PT23A	6	10.9726	PT23B	6	11.1054	PT23C	7	11.3193	PT23D	6	11.1945	PT231C	5	10.7388
PT232C	5	11.1039	PT233C	5	11.1151	PT234C	4	11.0385	PT235C	5	10.9861	PT236C	3	11.2288
PT237C	5	11.1826	T23A	5	581.988	T23b	5	581.548	PT2C*	24	10.7033	PS2C1	3	9.31393
PS2C0	3	8.64090	PS2C10	6	8.97742	T2C	20	577.807	PT2CA	5	10.5687	PT2C8	5	10.7766
PT2CC	5	10.7897	PT2C0	4	10.7040	PT2CE	5	10.6774	PT2C1	5	10.8642	PT2C2	5	10.9410
PT2C3	4	11.0547	PT2C4	5	10.7801	PT2C5	5	9.94662	T2CA	5	577.921	T2C3	5	576.755
T2CC	3	577.325	T2CD	4	580.468	T2CE	3	576.304	T2C1	4	578.199	T2C2	4	576.051
T2C3	4	575.707	T2C4	4	577.999	T2C5	4	581.079	PT3	19	61.5682	PS3H8	2	57.1132
PS3TP	1	58.7107	PS3HT	3	57.6457	T3	20	1019.20	PT3A	4	61.2715	PT3B	5	61.4783
PT3C	5	61.9979	PT3D	5	61.4660	PT31C	4	62.1541	PT32C	3	61.6732	PT33C	4	61.2054
PT34C	4	61.6371	PT35C	4	61.1976	T3A	5	1019.71	T3B	5	1019.71	T3C	5	1020.52
T3D	5	1016.68	T31C	4	1015.54	T32C	4	1014.03	T33C	4	1014.87	T34C	4	1022.02
T35C	4	1029.56	PT51	5	23.6214	T51	20	1907.95	T51A	3	1846.97	T51B	3	2100.28
T51C	5	1960.62	T51D	5	1943.23	T51E	4	1699.49	T511C	4	1881.23	T512C	5	1948.91
T513C	3	2067.45	T514C	4	1895.46	T515C	4	1776.33	PT56	5	9.65376	T56	19	1582.00
T56A	5	1488.50	T56B	4	1526.26	T56C	5	1676.97	T56D	5	1625.14	T561C	4	1528.02
T562C	3	1661.49	T563C	4	1674.76	T564C	4	1609.27	T565C	4	1456.37	PT25	15	10.3787
PS25	4	9.23406	T25	11	584.231	PT25A	4	10.6034	PT25B	4	10.1618	PT25C	3	10.3213
PT25D	4	10.4141	PT251C	4	10.1291	PT252C	4	10.5496	PT253C	3	10.5458	PT254C	4	10.3323
T25A	4	582.939	T25B	3	590.725	T25C	4	580.652	T251C	3	589.124	T252C	3	584.056
T253C	2	580.154	T254C	3	582.230	PS0	4	3.20455	PS0S	4	3.21508	NF	3	9393.85
NC	3	12702.2	FP	2	-52.7302	F1	3	2119.60	F2	3	2084.53	F	6	2102.06
PT2TOP	19	4.83603	PT2BOT	9	4.81595	PT9	20	6.03995						

Figure 1. - Concluded.

FIRST CLASS MAIL



POSTAGE AND FEES PAID
NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION

POSTMASTER: If Undeliverable (Section 158
Postal Manual) Do Not Return

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

— NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

**SCIENTIFIC AND TECHNICAL INFORMATION DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Washington, D.C. 20546**