NBS TECHNICAL NOTE

446

PRECISE: A Multiple Precision Version of Omnitab

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards

NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards[1] was established by an act of Congress March 3, 1901. Today, in addition to serving as the Nation's central measurement laboratory, the Bureau is a principal focal point in the Federal Government for assuring maximum application of the physical and engineering sciences to the advancement of technology in industry and commerce. To this end the Bureau conducts research and provides central national services in three broad program areas and provides central national services in a fourth. These are: (1) basic measurements and standards, (2) materials measurements and standards, (3) technological measurements and standards, and (4) transfer of technology.

The Bureau comprises the Institute for Basic Standards, the Institute for Materials Research, the Institute for Applied Technology, and the Center for Radiation Research.

THE INSTITUTE FOR BASIC STANDARDS provides the central basis within the United States of a complete and consistent system of physical measurement, coordinates that system with the measurement systems of other nations, and furnishes essential services leading to accurate and uniform physical measurements throughout the Nation's scientific community, industry, and commerce. The Institute consists of an Office of Standard Reference Data and a group of divisions organized by the following areas of science and engineering:

Applied Mathematics--Electricity--Metrology--Mechanics--Heat--Atomic Physics--Cryogenics[2]--Radio Physics[2]--Radio Engineering[2]--Astrophysics[2]--Time and Frequency.[2]

THE INSTITUTE FOR MATERIALS RESEARCH conducts materials research leading to methods, standards of measurement, and data needed by industry, commerce, educational institutions, and government. The Institute also provides advisory and research services to other government agencies. The Institute consists of an Office of Standard Reference Materials and a group of divisions organized by the following areas of materials research:

Analytical Chemistry--Polymers--Metallurgy--Inorganic Materials--Physical Chemistry.

THE INSTITUTE FOR APPLIED TECHNOLOGY provides for the creation of appropriate opportunities for the use and application of technology within the Federal Government and within the civilian sector of American industry. The primary functions of the Institute may be broadly classified as programs relating to technological measurements and standards and techniques for the transfer of technology. The Institute consists of a Clearinghouse for Scientific and Technical Information,[3] a Center for Computer Sciences and Technology, and a group of technical divisions and offices organized by the following fields of technology:

Building Research--Electronic Instrumentation--Technical Analysis--Product Evaluation--Invention and Innovation--Weights and Measures--Engineering Standards--Vehicle Systems Research.

THE CENTER FOR RADIATION RESEARCH engages in research, measurement, and application of radiation to the solution of Bureau mission problems and the problems of other agencies and institutions. The Center for Radiation Research consists of the following divisions:

Reactor Radiation--Linac Radiation--Applied Radiation--Nuclear Radiation.

---

[1]Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington, D.C. 20234.
[2]Located at Boulder, Colorado 80302.
[3]Located at 5285 Port Royal Road, Springfield, Virginia 22151.

UNITED STATES DEPARTMENT OF COMMERCE
C. R. Smith, Secretary
NATIONAL BUREAU OF STANDARDS • A. V. Astin, Director

NBS TECHNICAL NOTE 446

ISSUED JUNE 1968

PRECISE: A Multiple Precision Version of Omnitab

Alfred E. Beam
Computer Science Center
University of Maryland
College Park, Maryland 20742

and

Joseph Hilsenrath
Office of Standard Reference Data
Institute for Basic Standards
National Bureau of Standards
Washington, D.C. 20234

NBS Technical Notes are designed to supplement the Bureau's regular publications program. They provide a means for making available scientific data that are of transient or limited interest. Technical Notes may be listed or referred to in the open literature.

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C., 20402
Price 55 cents.

ABSTRACT

This users manual describes PRECISE - a completely assembled interpretive program for the IBM 7090/7094 which enables the user to carry out arithmetic operations and function generation in multiple precision (accuracy to 28 significant figures). PRECISE operates as a sub-monitor under the IBSYS or DC-IBSYS monitor systems. Appendixes describe how jobs are set up to be run under the PRECISE sub-monitor, and how the system may be expanded to include new subroutines. The program, which responds to instructions in the form of plain English sentences or contractions thereof, has provision for handling numbers out of the normal 7090/7094 range. It handles numbers as large as 10 to the 10 to the 9 power. Other features of the program include: free-field input; a work-sheet of 7,500 cells (3x2500 computer words) which can be dimensioned by the user at run time (75 rows by 100 columns, 300 rows by 25 columns, (etc.); solution of systems of linear equations in as many as 85 unknowns; flexible formatting; tape handling facility; and row and column sums. A description of the UOM Multiple Precision Package (SHARE Dist. No. 3081) is included as an appendix.

Key Words: Elementary functions, equation solver, double precision, linear equation solver, magnetic tape utility program, multiple-precision computing, multiple-precision programming, PRECISE, triple-precision, user's manual.

FOREWORD

The work which is reported here was started at the National Bureau of Standards and was completed at the University of Maryland after one of the authors (AEB) transferred to that Institution. The final version of PRECISE and of the Multiple Precision Package upon which it was built was prepared at the Computer Science Center of the University of Maryland and was supported in part be grant NsG-398 of the National Aeronautics and Space Administration.

The authors wish to acknowledge the help of Klaus Waibel in preparing portions of the text for computer-assisted printing and to Mrs. Bertha H. Walter for the painstaking work of checking the accuracy of many of the mathematical functions built into the PRECISE program.

A. V. Astin, Director

CONTENTS

1. INTRODUCTION

One of the more troublesome problems that confront the careful user of modern computers is the loss of significance resulting from round off and other computing pitfalls. In many calculations rounding errors are serious sources of annoyance -- in some they are downright fatal. While the recent trend to build computers with built-in hardware for double-precision operations is a decided help in this regard, the careful user of these features must still be on guard. He must be on guard for possible flaws in the hardware, or in the algorithms and even, unhappily, for errors in important constants used by the compiler or the conversion routines.

The problem has gotten worse recently as a consequence of the fact that many of the third generation computers have a shorter word length. As a result, programs which previously gave suitable answers in single precision now must be run in double precision.

We are tempted to speculate that if the cloak of anonymity were removed from commercial software systems and each subroutine or program or compiler segment were to carry the by-line of its author or authors, then, perhaps, there might be some improvement in the situation. But, be that as it may, there is a clear need for some yardsticks by which the accuracy of computer results can be judged. There is a need for a system which can deliver correct answers to a reasonably large number of significant digits even when handling exceedingly large or small numbers.

The release to SHARE in 1963 of a multiple precision package (UOM MPP SHARE DIST. NO. 3081) by Alfred E. Beam was a considerable boon to professional programmers using IBM 7090-94 computers. In spite of the existence of the MPP package and doubtless other similar packages, the problem of carrying out calculations in multiple precision involving the elementary trigonometric and transcendental functions is still by no means a trivial job. Nor is it easy even today to solve a large system of linear equations (in say 85 unknowns) and retain adequate accuracy.

Last place "errors" are so much a part of even reliable mathematical tables as to cause L. J. Comrie, a well-known table maker, to write a short piece entitled "What is an Error" (MTAC*, V.2, 1943, pp 284-286) in which he explains that when the seventh, eight, and ninth places in an entry in a mathematical table are 4,9,9 or 5,0,0, it matters little to the man who wants only seven places exactly what the tenth or eleventh place is. Thus, Comrie continues "...on more than one occasion I have written to our beloved editor saying 'I have found...errors of less than one unit in...tables, but am not sending them to you, lest you should be tempted to publish them."

Table makers are quite willing to accept these last-place or end figure "errors" because of the tedium of carrying out check calculations to three or more figures beyond those that they normally carry. PRECISE carries out calculations to many figures as a matter of course. Thus, there is really no need to tolerate "end-figure" errors.

Soon after it became clear that the philosophy behind, the organization and implementation of the OMNITAB general-purpose computing program on the 7094 was sound enough to attract a wide audience of problem solvers, whom even FORTRAN had not reached, we turned our attention for a time to the design of a comparable system for more precise calculations than were then possible in single precision. This system drew heavily on the multiple precision package designed by one of the authors to spare professional programmers the tedium of writing painstaking instructions for the computer to handle double and triple precision and out-of-range arithmetic. This report describes how the MPP package has been further employed to provide nonprogrammers with a computer tool for very precise calculations without the need to resort to conventional, and in this instance, very tedious programming.

---

* Mathematical Tables and Other Aids to Computation is now called "Mathematics of Computation."

The PRECISE program which is discussed here was designed to carry out arithmetic operations and function generation often to as many as 28 significant digits and at the very least to 21 figures. Except when instructed to increase the ranges, the program normally handles numbers x in the range 10exp(-76) to 10exp(76) and gives results to 28 significant figures. The program can also handle numbers outside of the above range. The greatest or smallest power of 10 can be as high as plus or minus one billion. In this extreme case the results are good only to 21 significant figures.

PRECISE, like its predecessor, OMNITAB*, is designed to provide a close parallel to the modus operandi in carrying out calculations with a desk calculator and a multi-columned (and multi-lined) worksheet. While the worksheet in OMNITAB for the 7094 was fixed at 101 rows by 46 columns, the 7500 cells (3 x 2500 computer words) set aside for the worksheet in PRECISE can be dimensioned at the start of each problem at run time.

---

*J. Hilsenrath, G. G. Ziegler, C. G. Messina, P. J. Walsh, and R. J. Herbold, OMNITAB - A Computer Program For Statistical and Numerical Analysis, National Bureau of Standards, Handbook 101, U. S. Government Printing Office, Washington, D. C. 20402 (March 4, 1966).

2. THE PRECISE PROGRAM

PRECISE is a completely assembled interpretive program for the IBM 7090/7094 which enables the user to carry out arithmetic operations and function generation - in multiple precision (accuracy to 28 significant figures). PRECISE operates as a sub-monitor under the IBSYS or DC-IBSYS monitor systems. Appendix I describes how jobs are set up to be run under the PRECISE sub-monitor. The program, which responds to instructions in the form of plain English sentences or contractions thereof, has provision for handling numbers out of the normal IBM 7090/7094 range. It handles numbers as large as 10 to the 10 to the 9 power. Other features of the program include:

a. free field input
b. a work-space of about 7500 cells which is normally set to simulate a work-sheet of 200 lines by 37 columns but can be redimensioned by the command NUMBER OF ROWS IS xx, MAXIMUM COLS IS xx, or simply by NUMBER 1000,7 for a work-sheet of 1000 lines and 7 columns
c. A simultaneous linear equation solver which can handle as many as 85 equations with full precision
d. format free output permitting printing and punching of results either in fixed or floating mode with 9, 14, 17, 25, 28 figures decimals as the case may be
e. formatting provision to provide a wide variety of output corresponding to the more common practice in published mathematical tables
f. provision for column headings of up to 18 characters
g. provision for taking sums of rows and of columns affords a moderately efficient means for carrying out numerical integration - with Gaussian or Lagrangian coefficients
h. tape handling facilities

The over-all logic and command structure is similar to that of the OMNITAB[1] program. There are however the following differences

a. the "work-sheet" can be dimensioned (the product of rows by columns cannot however exceed 7500)
b. format statements permit more sophisticated control

c. the program is normally set to handle numbers in the range 10^{-76} to 10^{76} but the range can be extended by the command GREATEST POWER OF TEN IS xx, where xx may be as large as 10^9

d. the first use of the command SUMMARIZE provides a column sum for every column printed; the second use turns the provision off, etc.

The program was written in FAP (or MAP) and makes use of an arithmetic package called MULTIPLE PRECISION PACKAGE UOM MPP for the IBM 7090/7094, developed by Alfred E. Beam (distributed to SHARE as Dist No 3081). The PRECISE program responds to commands such as:

a. GENERATE ARGS .0(.01)1.(1.)50. IN COL 8
b. READ COL 1,2,4,7
c. sub 1. from col 11 store in col 5
D. DIVIDE COL 21 BY 5040. MULT BY COL 7 ADD TO COL 8
e. FLOATING WITH 28 SIGNIFICANT FIGURES
f. LOGE OF COL 3 MULT BY -1. ADD TO COL 5
g. NEGEXP COL 1 STORE IN COL 2
h. PRINT COL 1,4,8,16

The logic of the command structure is such that the program also interprets the above instructions correctly even if all of the words but the first are omitted. Thus the above instructions can also be written as follows:

```
GENERA .0(.01)1.(1.)50.  8
READ 1,2,4,7
SUB 1., 11, 5
DIVIDE 21,5040., 7, 8
FLOATING 28
LOGE 3, -1., 5
NEGEXP 1, 2
PRINT 1 4 8 16
```

How this is achieved will be discussed later. At this point it should be observed that the presence of a decimal point in a number, identifies it as a particular number. The absence of a decimal point identifies it as a column number. The input is free field, the numbers need only be separated by one or more spaces, by a comma, or any non-numeric character other than +,-,E and the decimal point. See Figure 1 for examples of the variety of mixed data formats which PRECISE accepts.

The following arithmetic and mathematical operations are provided.

ADD	SUBTRACT	MULTIPLY	DIVIDE
NEGATIVE	SUB	MULT	DIV
SQRT	RAISE	RAISEI	ABSOLUTE
LOGE	LOGTEN	EXP	NEGEXP
SIN	COS	TAN	COT
SIND	COSD	TAND	COTD
SINH	COSH	TANH	COTH
ARCSIN	ARCCOS	ARCTAN	ARCCOT
ASIND	ACOSD	ATAND	ACOTD
ASINH	ACOSH	ATANH	ACOTH

The arithmetic operations and function generating facilities of PRECISE are augmented by a flexible input and a number of output option which permit one to print results to 9, 14, 17, 25, or 28 figures in either fixed point or floating point without writing format statements. Where more flexibility is needed than is afforded by the built-in formats, the program recognizes and interprets format statements providing considerably more flexibility than is available in FORTRAN.

Extensive tape handling facilities have been built into PRECISE to provide a back-up store when the problem begins to tax the work-space of 7500 cells (3x7500 computer words). The system provides for mounting and loading of tapes; for setting the read and write density; for positioning tapes and skipping files and records (in both directions). These tape manipulation facilities permit one to transfer (PUT) data from the core (work sheet) in binary form and to retrieve (GET) them again in the same run or in subsequent runs. The program can read from as many as five tape units and punch out print on an equal number. Input tapes can be switched during a run between any of six logical tape unit numbers (2, 3, 4, 5, 9, or 10).

As the majority of the problems which PRECISE was designed to handle require a moderately small amount of input data, the program automatically writes each instruction, after its execution, on a scratch tape and then prints the instructions at the end of the calculation. Where the program is applied to a problem in which the input data involve hundreds or even thousands of cards and the automatic listing of input is not deemed important, it can be suppressed by the instruction NOLIST. The listing of input can be restored again with the word LIST.

In the normal mode of operation, each time the PRINT instruction is used, a new page is called. Also column headings are supplied over each of the columns printed. There are cases, however, when the user would prefer not to start a new page and not to have column headings. Provision is, therefore, made to suppress these by DONTPAGE and DONTHEAD. Here again these provisions can be restored by the use of the instructions DOPAGE and DOHEAD.

As the use of tabular data is enhanced by arranging them in blocks of 10 lines separated by a blank line, this provision is built into the program. Again, if this feature is not wanted, it can be suppressed by the instruction DONTSPACE and reinstated by DOSPACE.

```
   SOURCE PROGRAM LISTING

 PRECISE
 TITLE ELEMENTARY AND TRANSCENDENTAL FUNCTIONS TO 28 FIGURES
 FLOATING WITH 28 SIGNIFICANT FIGURES
 GENERATE .01(.01).1(.1)1.(1.)10.(10.)100.(100.)1000. IN  1
 SIN OF 1 STORE IN 2
 DIVIDE 1.0 BY 2 AND STORE IN 4
 HEADING ON 1 CARD
     X                    SIN(X)
 PRINT 1,2
 HEADING ON 1 CARD
     X
 PRINT   ON 1 CARD
     X                    COT(X)
 PRINT 1,2
 STOP

END OF SOURCE PROGRAM LISTING.

    356 LINES OUTPUT.
```

3. PRECISE VOCABULARY, LOGICAL STRUCTURE AND OPERATIONS

```
 + = A COLUMN NUMBER
 * = A FLOATING POINT NUMBER
 $ = A FIXED POINT CONSTANT(INTEGER)
 $$= A COLUMN NUMBER OR A FLOATING POINT CONSTANT

   INPUT OPERATIONS
*****************************************************************
 PRECISE     ANY ADDITIONAL WORDS IN CARD COLUMNS 13-72
 REMARK      ANY COMMENT IN CARD COLUMNS 7-72
 TITLE       ANY INFORMATION IN CARD COLUMNS 7-72
 GENERATE    ARGUMENTS *(*)*(*)...*(*)* IN COLUMN +
 READ        COLUMNS +,+,...., AND +
 STOP        ENDS A BATCH OF PROGRAMS
*****************************************************************

   OUTPUT OPERATIONS
*****************************************************************
 FIXED       FORMAT WITH $ FIGURES TO RIGHT OF DECIMAL POINT
 FLOATING    FORMAT WITH $ SIGNIFICANT FIGURES
 HEAD        COLUMN + WITH CHARACTERS IN COLUMNS 55-72 OF THIS CARD
 HEADING     ON $ CARDS WHICH MUST FOLLOW
 FORMAT      ON $ CARDS WHICH MUST FOLLOW
 PRINT       COLUMNS +,+,...., AND +
 PUNCH       COLUMNS +,+,...., AND +
*****************************************************************

   MODIFICATION OPERATIONS
*****************************************************************
 GREATEST POWER OF TEN NEEDED WILL BE $ (LESS THAN TEN TO NINTH)
 NUMBER      OF ROWS = $ AND MAXIMUM COLUMN NUMBER = $
 SUMMARIZE
*****************************************************************

   INTERNAL OPERATIONS
*****************************************************************
 ROWSUM      $$,$$,....,+,+,....,+,$$ AND STORE IN COLUMN +
 RAISEI      RAISE $$ TO $ POWER AND STORE IN COLUMN +
 SOLVE       THE FOLLOWING SYSTEM OF $ LINEAR EQUATIONS
 CHANGE      THE SIGN OF COLUMN +
 ERASE       COLUMNS +,+,...., AND +
 ADD         $$ TO $$ AND STORE IN COLUMN +
 SUBTRACT    $$ FROM $$ AND STORE IN COLUMN +
 MULTIPLY    $$ BY $$ AND STORE IN COLUMN +
 DIVIDE      $$ BY $$ AND STORE IN COLUMN +
 RAISE       $$ TO THE $$ POWER AND STORE IN COLUMN +
 ABSOLUTE    VALUE OF $$ AND STORE IN COLUMN +
 ACOSD       OF $$ AND STORE IN COLUMN + (ARC COSINE--DEGREES)
 ACOSH       OF $$ AND STORE IN COLUMN + (HYPERBOLIC COSINE)
 ACOTD       OF $$ AND STORE IN COLUMN + (ARC COTANGENT--DEGREES)
 ACOTH       OF $$ AND STORE IN COLUMN + (HYPERBOLIC ARC COTANGENT)
 ARCCOS      OF $$ AND STORE IN COLUMN + (ARC COSINE--RADIANS)
 ARCCOT      OF $$ AND STORE IN COLUMN + (ARC COTANGENT--RADIANS)
 ARCSIN      OF $$ AND STORE IN COLUMN + (ARC SINE--RADIANS)
 ARCTAN      OF $$ AND STORE IN COLUMN + (ARC TANGENT--RADIANS)
 ASIND       OF $$ AND STORE IN COLUMN + (ARC SINE--DEGREES)
```

ASINH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC ARC SINE)
ATAND	OF $$ AND STORE IN COLUMN + (ARC TANGENT--DEGREES)
ATANH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC ARC TANGENT)
COS	OF $$ AND STORE IN COLUMN + (COSINE--RADIANS)
COSD	OF $$ AND STORE IN COLUMN + (COSINE--DEGREES)
COSH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC COSINE)
COT	OF $$ AND STORE IN COLUMN + (COTANGENT--RADIANS)
COTD	OF $$ AND STORE IN COLUMN + (COTANGENT--DEGREES)
COTH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC COTANGENT)
EXP	OF $$ AND STORE IN COLUMN + (EXPONENTIAL FUNCTION)
LOGE	OF $$ AND STORE IN COLUMN + (NATURAL LOGARITHM)
LOGTEN	OF $$ AND STORE IN COLUMN + (COMMON LOGARITHM)
NEGATIVE	OF $$ AND STORE IN COLUMN + (CHANGES SIGN AND STORES)
NEGEXP	OF $$ AND STORE IN COLUMN + (NEGATIVE EXPONENTIAL)
SIN	OF $$ AND STORE IN COLUMN + (SINE--RADIANS)
SIND	OF $$ AND STORE IN COLUMN + (SINE--DEGREES)
SINH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC SINE)
SQRT	OF $$ AND STORE IN COLUMN + (SQUARE ROOT)
TAN	OF $$ AND STORE IN COLUMN + (TANGENT--RADIANS)
TAND	OF $$ AND STORE IN COLUMN + (TANGENT--DEGREES)
TANH	OF $$ AND STORE IN COLUMN + (HYPERBOLIC TANGENT)
ADD	$$ TO $$, MULTIPLY BY $$, AND ADD TO COLUMN +
SUBTRACT	$$ FROM $$, MULTIPLY BY $$, AND ADD TO COLUMN +
MULTIPLY	$$ BY $$, MULTIPLY BY $$, AND ADD TO COLUMN +
DIVIDE	$$ BY $$, MULTIPLY BY $$, AND ADD TO COLUMN +
RAISE	$$ TO $$ POWER, MULTIPLY BY $$ AND ADD TO COLUMN +
ABSOLUTE	VALUE OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ACOSD	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ACOSH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ACOTD	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ACOTH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ARCCOS	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ARCCOT	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ARCSIN	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ARCTAN	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ASIND	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ASINH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ATAND	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
ATANH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COS	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COSD	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COSH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COT	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COTD	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
COTH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
EXP	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
LOGE	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
LOGTEN	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
NEGATIVE	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
NEGEXP	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
SIN	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
SIND	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
SINH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
SQRT	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
TAN	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
TAND	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +
TANH	OF $$, MULTIPLY BY $$ AND ADD TO COLUMN +

```
***********************************************************************

  TAPE MANIPULATION OPERATIONS($ MAY BE LOGICAL UNIT 2,3,4,9, OR 10)
***********************************************************************
 REWIND     TAPES $,$,....,$
 UNLOAD     TAPES $,$,....,$
 SETLOW     DENSITY ON TAPES $,$,....,$
 SETHIGH    DENSITY ON TAPES $,$,....,$
 ENDFILE    TAPES $,$,....,$
 POSITION   TAPE $,$ FILES AND $ RECORDS BACKWARDS, THEN $ FILES
                  AND $ RECORDS FORWARDS
***********************************************************************

  BINARY TAPE INPUT/OUTPUT OPERATIONS($ MAY BE UNIT 2,3,4,9, OR 10)
***********************************************************************
 PUT        ON TAPE $, COLUMNS +,+,....,+
 GET        FROM TAPE $, COLUMNS +,+,....,+
***********************************************************************

  OTHER BCD TAPE OPERATIONS
***********************************************************************
 PRNT2      COLUMNS +,....,+ (SAME AS PRINT EXCEPT TAPE  2 IS WRITTEN)
 PRNT3      COLUMNS +,....,+ (SAME AS PRINT EXCEPT TAPE  3 IS WRITTEN)
 PRNT4      COLUMNS +,....,+ (SAME AS PRINT EXCEPT TAPE  4 IS WRITTEN)
 PRNT9      COLUMNS +,....,+ (SAME AS PRINT EXCEPT TAPE  9 IS WRITTEN)
 PRNT10     COLUMNS +,....,+ (SAME AS PRINT EXCEPT TAPE 10 IS WRITTEN)
 PNCH2      COLUMNS +,....,+ (SAME AS PUNCH EXCEPT TAPE  2 IS WRITTEN)
 PNCH3      COLUMNS +,....,+ (SAME AS PUNCH EXCEPT TAPE  3 IS WRITTEN)
 PNCH4      COLUMNS +,....,+ (SAME AS PUNCH EXCEPT TAPE  4 IS WRITTEN)
 PNCH9      COLUMNS +,....,+ (SAME AS PUNCH EXCEPT TAPE  9 IS WRITTEN)
 PNCH10     COLUMNS +,....,+ (SAME AS PUNCH EXCEPT TAPE 10 IS WRITTEN)
 READ2      COLUMNS +,....,+ (SAME AS READ EXCEPT DATA IS FROM TAPE  2)
 READ3      COLUMNS +,....,+ (SAME AS READ EXCEPT DATA IS FROM TAPE  3)
 READ4      COLUMNS +,....,+ (SAME AS READ EXCEPT DATA IS FROM TAPE  4)
 READ9      COLUMNS +,....,+ (SAME AS READ EXCEPT DATA IS FROM TAPE  9)
 READ10     COLUMNS +,....,+ (SAME AS READ EXCEPT DATA IS FROM TAPE 10)
 SWITCH     INPUT TAPE TO BE LOGICAL TAPE $(MAY BE 2,3,4,5,9, OR 10)
 TRANSFER   FOLLOWING DATA TO LOGICAL TAPE $(MAY BE 2,3,4,9, OR 10)
 ENDTRANSFER OF DATA(TERMINATES THE TRANSFER OPERATION)
***********************************************************************

  OUTPUT CONTROL OPERATIONS
***********************************************************************
 NOLIST     SUPPRESSES LISTING OF THE SOURCE PROGRAM
 LIST       RESTORES LISTING OF THE SOURCE PROGRAM
 DONTHEAD   SUPPRESSES PAGE HEADING DURING THE PRINT OPERATION
 DOHEAD     RESTORES PAGE HEADING DURING THE PRINT OPERATION
 DONTPAGE   SUPPRESSES BEGINNING A NEW PAGE
 DOPAGE     RESTORES BEGINNING OF NEW PAGES
 DONTSPACE  SUPPRESSES BLANK LINES BETWEEN TEN LINE BLOCKS
 DOSPACE    RESTORES BLANK LINES BETWEEN TEN LINE BLOCKS
***********************************************************************
```

4. DESCRIPTION OF OPERATIONS (GENERAL)

PRECISE: This card must be first in each program. Card columns 13-72 are placed in the page line which heads each page unless it is overwritten by use of the TITLE card.

REMARK: Card columns 7-72 are printed on the output tape at the time this card is read.

TITLE: Card columns 7-72 of this card are inserted into the page line.

GENERATE: A column is generated as specified and the number of rows is reset to be the number of arguments generated.

READ: Data is read into specified columns, one row per card and the number of rows is reset to be the number of data cards read.

STOP: This card must be the last of a set of PRECISE programs and is the signal to send control to the monitor system.

FIXED: The print format is set to be fixed and the number of decimals to the right of the decimal point is set to be 9, 14, 17, 25, or 28, as specified.

FLOATING: The print format is set to be floating and the number of decimals to the right of the decimal point is set to be 9, 14, 17, 25, or 28 as specified.

Table A summarizes the output options which are available by the use of the FIXED and FLOATING commands. The first format in the table is the one which is set initially, and it remains in force until a FIXED, FLOATING, or FORMAT command is encountered. APPENDIX II gives a general description of formats.

HEAD: An 18 character heading for the specified Column is taken from card columns 55-72 of the HEAD card. Provision was made to head only columns 1-63 so one should not print a column whose number is greater than 63 unless the HEADING card has been used.

4

HEADING: The heading line is blanked and the first 72 characters are replaced by card columns 1-72 of the first card following the HEADING. If 2 heading cards are specified then heading line characters 73-120 are replaced by card columns 1-48 of the second card following the HEADING card.

FORMAT: The format area is filled with card columns 1-72 of the specified (1 to 4) number of cards which follow the FORMAT card. Blanks are not allowed except in Hollerith fields and the format must begin in card column 1 of the first card. The first blank not in a Hollerith field terminates the format. A format description is given in APPENDIX II.

PRINT: The specified columns are written on the regular output tape according to the current format.

PUNCH: The specified columns are written on the regular punch tape according to the current format.

GREATEST: The parameter $ = P causes the entire package to be set to handle numbers (N) in the range 10**-P<|N|<10**P If this operation is not specified then P = 76. If given, it must be given before any computation, column input or output. P can be increased by a following program but P cannot be decreased within the same job. In the initial condition when p=76, computation is carried out using 28 significant decimal digits. When p is increased, the accuracy is decreased. The approximate number of significant decimal digits in effect is 30-Q where Q is the closest power of ten for p. Since p may be set to any value in the range from 76 to 1 billion, the accuracy of calculations will be from 28 to 21 significant decimal digits.

NUMBER: The row and column parameters can be set at any time but the product of the two dimensions must not exceed the matrix storage capacity.

SUMMARIZE: The column summing switch is turned on if off, or off if on. If the summing switch is on when a PRINT is executed then the sum of the printed columns will also be printed.

5. DESCRIPTION OF THE INTERNAL OPERATIONS

For description of the internal operations, use is made of the following definitions:

1) k is the current column length.
2) n is the number of arguments specified on a card.
3) C(i) is the ith argument, i = 1,2,...,n.
4) [C(i)] is the column of numbers specified by C(i) If C(i) is not a column number, [C(i)] can be thought of as a column of numbers with all elements equal to C(i)
5) Algebraic symbols are used to indicate operations repeated for each element of the column.
6) [W]<--[Z] means "column W is replaced by column Z".

The table below gives an operation name followed by a description of the function of that particular operation.

GROUP I

ROWSUM	[C(n)]<--[C(1)]+...+[C(n-1)]
RAISEI	[C(n)]<--[C(1)]**C(2), C(2) must be an integer
CHANGE	[C(2)]<--[C(1)]
ERASE	[C(j)]<--0, j = 1,2,...,n
SOLVE	This operation must be followed by C(1){C(1)+1} numbers punched (starting in column 2) on as many cards as necessary. Each card must have the character D punched in column 1. Numbers are of floating point type and are separated by commas. The first blank encountered terminates the card. An example of the operation and data for solution of the two linear equations -3x + 9y = -6 -5x - y = 4 is as follows: SOLVE 2 EQUATIONS D-3.E0,9.0,-6 D-00.05E2,-1.0 D.00004E+5 All of the above data could have been punched on one card. The matrix and solutions are printed and then all columns are cleared before proceeding to the next operation.

GROUP II (BINARY OPERATORS) *

```
ADD         [C(n)]<--[C(1)]+[C(2)]
SUBTRACT    [C(n)]<--[C(2)]-[C(1)]
MULTIPLY    [C(n)]<--[C(1)]*[C(2)]
DIVIDE      [C(n)]<--[C(1)]/[C(2)]
RAISE       [C(n)]<--[C(1)]**[C(2)]
```

The 5 operations above require that the number of arguments n be 3. There is a corresponding operation for each of these when an additional argument is specified.

Let [F] be the resultant column for any of the above operations. Then the column result [C(n)] for an additional argument can be described as

```
[C(n)]<--[C(n)]+[F]*[C(n-1)]
```

*Note: DIV may be used rather than DIVIDE, MULT or MPY may be used rather than MULTIPLY, and SUB may be used rather than SUBTRACT.

GROUP III (UNARY OPERATORS)

ABSOLUTE	[C(n)]<--$\lvert$[C(1)]$\rvert$
NEGATIVE	[C(n)]<-- -[C(1)]
EXP	[C(n)]<--e**[C(1)]
NEGEXP	[C(n)]<--e**-[C(1)]
LOGE	[C(n)]<--log [C(1)]
LOGTEN	[C(n)]<--common logarithm of [C(1)]
SQRT	[C(n)]<--the square root of [C(1)]
SIN	[C(n)]<--sin[C(1)], radian argument
SIND	[C(n)]<--sin[C(1)], degree argument
COS	[C(n)]<--cos[C(1)], radian argument
COSD	[C(n)]<--cos[C(1)], degree argument
TAN	[C(n)]<--sin[C(1)]/cos[C(1)], radian argument
TAND	[C(n)]<--sin[C(1)]/cos[C(1)], degree argument
COT	[C(n)]<--cos[C(1)]/sin[C(1)], radian argument
COTD	[C(n)]<--cos[C(1)]/sin[C(1)], degree argument
ARCTAN	[C(n)]<--tan^{-1}[C(1)], result in radians
ATAND	[C(n)]<--tan^{-1}[C(1)], result in degrees
ARCCOS	[C(n)]<--cos^{-1}[C(1)], result in radians
ACOSD	[C(n)]<--cos^{-1}[C(1)], result in degrees
ARCSIN	[C(n)]<--sin^{-1}[C(1)], result in radians
ASIND	[C(n)]<--sin^{-1}[C(1)], result in degrees
ARCCOT	[C(n)]<--cot^{-1}[C(1)], result in radians
ACOTD	[C(n)]<--cot^{-1}[C(1)], result in degrees
SINH	[C(n)]<--sinh[C(1)]
COSH	[C(n)]<--cosh[C(1)]
TANH	[C(n)]<--tanh[C(1)]
COTH	[C(n)]<--coth[C(1)] = 1/tanh[C(1)]
ASINH	[C(n)]<--sinh^{-1}[C(1)]
ACOSH	[C(n)]<--cosh^{-1}[C(1)]
ATANH	[C(n)]<--tanh^{-1}[C(1)]
ACOTH	[C(n)]<--coth^{-1}[C(1)]

The above operations (Group III) require that the number of arguments n be 2. There is a corresponding operation for each of these when an additional argument is specified.

Let [F] be the resultant column for any of the above operations. Then the column result [C(n)] for an additional argument can be described as

[C(n)]<--[C(n)]+[F]*[C(n-1)]

6. BINARY TAPE OPERATIONS AND TAPE MANIPULATION

The following operations are designed for positioning of tapes, writing tape marks, saving of temporary results, and for reading of data which was previously written on tape. In the descriptions T means a logical tape number and T may have integral values equal to 2,3,4,9, or 10.

<u>REWIND TAPES T(1),T(2),...</u>

This operation causes all tapes specified to be rewound.

<u>UNLOAD TAPES T(1),T(2),...</u>

This operation causes all tapes specified to be rewound and unloaded.

<u>SETLOW DENSITY ON TAPES T(1),T(2),...</u>

This operation causes all tapes specified to be set to low density

<u>SETHIGH DENSITY ON TAPES T(1),T(2),...</u>

This operation causes all tapes specified to be set to high density.

<u>ENDFILE TAPES T(1),T(2),...</u>

This operation causes a tape mark (end of file) to be written on all specified tapes.

<u>POSITION TAPE T,a,b,c,d</u>

This operation causes tape T to be positioned as follows:

1) Tape T is moved backwards over <u>a</u> tape marks.
2) Tape T is moved backwards over <u>b</u> records.
3) Tape T is moved forward over <u>c</u> tape marks.
4) Tape T is moved forward over <u>d</u> records.

Any of the numbers a,b,c, or d may be zero, but may not be omitted from the operation.

<u>PUT ON TAPE T, COLUMNS C(1),C(2),...,C(n)</u>

This operation causes all specified columns Ci (1 record per column) to be written on tape T. If no Ci are specified then the entire working area is written as one record on tape T.

GET FROM TAPE T, COLUMNS C(1),C(2),...,C(n)

This operation causes all specified columns (1 record per column) to be replaced by columns obtained from tape T. If no Ci are specified then the entire working area is replaced by the next record from tape T. All information read via the 'GET' operation should have been put on the tape via the 'PUT' OPERATION.

OTHER BCD TAPE OPERATIONS

The following operations are mainly useful for special purpose jobs, such as building or reading special input tapes or preparing special output tapes. In the description of the operation T is an integral logical tape number and the only legal values for T are 2,3,4,9, or 10. Operation names terminating with i means that the operation name is defined for i = 2,3,4,9, or 10 depending on the value of i.

PRNTi COLUMNS C(1),C(2),...,C(n)

This operation is the same as 'PRINT' except the specified columns are printed on logical tape i rather than the regular output tape.

PNCHi COLUMNS C(1),C(2),...,C(n)

This operation is the same as 'PUNCH' except the specified columns are printed on logical tape i rather than the regular punch tape.

READi COLUMNS C(1),C(2),...,C(n)

This operation is the same as 'READ' except the specified columns are read from tape i rather than the regular input tape.

SWITCH INPUT TAPE TO BE T

This operation causes the next operation or data card (and following cards) to be read from tape T rather than the regular input tape.

TRANSFER NEXT STATEMENTS TO TAPE T

This operation causes all following cards to be transfered to tape T without any attempt to interpret the cards. All cards up to, but not including, a 'ENDTRA' operation will be transfered.

ENDTRANSFER OF INSTRUCTIONS

This operation terminates the transfer of cards to another tape which was initiated by the 'TRANSFER' operation.

OUTPUT CONTROL OPERATIONS

The following operations give the user some control over the heading, spacing, and program listing.

NOLIST

This operation suppresses listing of the PRECISE source program.

LIST

This operation restores the listing of the PRECISE source program. LIST is initially the mode of listing.

DONTHEAD

This operation suppresses heading of a page when the PRINT operation is encountered.

DOHEAD

This operation restores heading of pages. This is the initial mode.

DONTPAGE

This operation suppresses the beginning of a new page when the PRINT operation is encountered.

DOPAGE

This operation restores the paging of output, and is the initial mode.

DONTSPACE

This operation suppresses the printing of a blank line between each block of ten lines.

DOSPACE

This operation restores the printing of a blank line between each block of ten lines. This is the initial mode.

7. RESTRICTIONS ON THE OPERATIONS

1) The operation word must be the first character punched on a card.

2) Except for the PRECISE, REMARK, HEAD, TITLE, FORMAT data, and D-type data cards; spacing is arbitrary, anywhere within the 72 columns of a card. PRECISE, REMARK, HEAD, and TITLE should be punched starting in card column 1.

3) All operation words must be at least 6 characters long or separated from following information by a blank.

4) An argument Ci is composed of a continuous string from the set of characters 0,1,...,9, decimal point, E, +, or -. Any character(s) other than this set acts as an argument separator. Ci will be a multiple precision number if either of the characters E or . appear in the field. Ci will be an integer if neither E nor . appear.

5) In most operations an integer argument must be less than or equal to the maximum column number allowable. Exceptions to this rule are GENERATE, GREATEST, NUMBER, FORMAT, FIXED, FLOATING, HEAD, HEADING, SUMMARIZE, SOLVE, REMARK, TITLE, RAISEI, REWIND, UNLOAD, SETLOW, SETHIGH, ENDFILE, PUT, GET, and POSITION.

ARG	TOTAL COLUMN WIDTH PER NUMBER	MAXIMUM NUMBER OF COLUMNS PRINTABLE	MAXIMUM NUMBER OF COLUMNS PUNCHABLE	FLOATING FORMAT	FIXED FORMAT
9	18	7	4	0K1S1P5(7E9.18	1X0K1P5(7F9.18
14	24	5	3	0K1S1P5(5E14.24	1X0K1P5(5F14.24
17	30	4	2	6K0S1P8(4E17.30	6K0S1P8(4F17.30
25	42	3	1	5K1S1P9(3E25.42	5K0S1P9(3F25.42
28	42	3	1	5K1P6(3E28.42	5K1P6(3F28.42

Table A. Output options provided by various built-in formats

8. ERRORS AND ERROR DETECTION

The following are considered errors by PRECISE, and cause execution of a program to be terminated whenever such an error is detected.

1) An illegal operation word.
2) A column number which is too large.
3) Illegal data.
4) Overflow during computation or input conversion. Overflow occurs when a number is computed which is too large to be represented within the currently defined binary number format.
5) Division by zero.
6) Output FORMAT errors.

Whenever one of the above errors is detected, one or more messages are printed, and execution of the rest of the PRECISE commands is terminated; however, the remainder of the commands are scanned to determine legality of the operation words.

```
PRECISE      HILSENRATH-BEAM    CHECK1--I/O AND GENERATE
REMARK 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ*()/$-+ = REMARK CARD XXXXXXX
 READ COLUMNS 1,2,3, AND 4 MAXIMUM NO ROWS IS TWO HUNDRED AT THIS POINT
   .000000000001 1000000000000000000000000. -1   178.624
  -.000000000011  110000000000000000000000.  2.     8.0006235E1
   .00000000012    12000000000000000000000.  3    -7.21E-4
  -.0000000013      1300000000000000000000. -4 3.14159E21
   .000000014E-0      140000000000000000000.  5              -.00624E0001
  -.00000015           15000000000000000000. -6.     527.11111111E0
   .0000016F+0          1600000000000000000.  7    000000.00000
  -.000017               170000000000000000.  8. 0000000000000000000000000001
   .00018                 18000000000000000.  9  1000000000000000000000000E10
  -.0019E000               1900000000000000.  0  999999999999999999999999999.
 87654321.4E-54 -.0000000000987654321E-36   .0 .222222222222222222222222222
 PRINT COLUMNS 1,2,3,AND 4 FORMAT IS FLOATING WITH NINE S AUTOMATICALLY
 TITLE REPLACES INFO THAT WAS ON THE PRECISE CARD XXXXXXXXXXXXXXXXXXXXX
 FLOATING WITH 28 SIG FIG
 PRINT COLUMNS 4 AND 1
 FLOATING WITH 25 SIG FIG                        THREE COLUMN LIMIT
 PRINT COLUMNS 4,1, AND 2
 FLOATING WITH 20 SIG FIG                        FOUR  COLUMN LIMIT
 PRINT COLS 4,1,2,3
 FLOATING WITH 14 SIG FIG                        FIVE COLUMN LIMIT
 PRINT COLS 4,1,2,3 AND REPEAT 4 AGAIN
 ROWSUM COLUMNS 4,3,2,1,1 TO COLUMN 6
 SUMMARIZE    THIS CARD SETS SUMMING TRIGGER ON
 FLOATING WITH  9 SIGFIG                        SIX COLUMN LIMIT
 PRINT 1,2,3,4,1,6
 GENERATE ARGUMENTS 0(1.)40.                    INTO COLUMN 1
GENERATE ARGUMENTS 0.(1.)5.(1.)10.(1.)30.(1.)40.   INTO COLUMN 2
 GENERATE ARGUMENTS40.(-1.0)0.0                 INTO COLUMN 3
 GENERATE ARGUMENTS .1(.1)1.(.1)4.1             INTO COLUMN 5
 PRINT 1,2,3,4,5,AND 6                          *******  END OF TEST ONE
```

Fig. 1. A test problem showing free-field data input and typical I/O instructions.

```
      PRECISE
      TITLE ELEMENTARY ANC TRANSCENDENTAL FUNCTIONS TO 25 FIGURES
      FORMAT CN 1 CARD
      1C 1F2.9,1P5K1S7(1E25.45
      GENERATE .C1(.01).1(.1)1.(1.)1C.(10.)1CC.(100.)1000. IN COL 1
      GREATEST PCWER CF TEN IS 10000
      ASINH 1,2
      HEADING CN 1 CARD
            X                          ASINH(X)
      PRINT 1,2
      ACCSH 1,2
      HEADING CN 1 CARD
            X                          ACOSH(X)
      PRINT 1,2
      ATANH 1,2
 **$  EITHER AN CVERFLCW CR AN UNDEFINED FUNCTION HAS BEEN ENCOUNTERED.
   *  OVERFLCW MAY TAKE PLACE ANYWHERE--INCLUDING INPUT-OUTPUT.  A RERUN WITH
   *  THE PCWER CF TEN CAPACITY INCREASED COULD CLEAR UP OVERFLOW.
   *  AN UNDEFINED FUNCTICN MAY BE DIVIDE,ACOTH,ATANH,TAN,TAND,COTH,COT,COTD,
   *  LOGE, CR LCGTEN.  THE REST OF THE JOB WILL BE SKIPPED.
      HEADING CN 1 CARD
 ***        X                          ATANH(X)
      PRINT 1,2
      ACOTH 1,2
      HEADING CN 1 CARD
 ***        X                          ACOTH(X)
      PRINT 1,2
```

Figure 2. A typical source program listing showing some diagnostic statements.

```
PRECISE
REMARK  THIS IS PART OF A TEST OF THE EQUATION
REMARK  SOLVER PORTION OF PRECISE TO CHECK THE
REMARK  INFLUENCE OF THE MULTIPLE PRECISION
REMARK  OPERATIONS WHEN APPLIED TO CERTAIN
REMARK  ILL-CONDITIONED MATRICES
TITLE   TEST OF OPERATIONS ON THE HILBERT MATRIX I=4
SOLVE  THE FOLLOWING SYSTEM OF 4 EQUATIONS
D1.0,0.5,0.33333333333333333333.0.25
D2.0833333333333333333
D0.5,0.33333333333333333333,0.25 0.2
D1.2833333333333333333
D0.33333333333333333333,0.25,0.2,0.16666666666666666667
D0.95
D0.25,0.2,0.16666666666666666667,0.14285714285714285714
D0.7595238095238095238095
PRECISE TEST OF OPERATIONS ON THE HILBERT MATRIX I=3
SOLVE THE FOLLOWING SYSTEM OF 3 EQUATIONS
D1.0,0.5,0.33333333333333333333,1.8333333333333333333
D0.5,0.33333333333333333333,0.25,1.08333333333333333333
D0.33333333333333333333,0.25,0.2,0.78333333333333333333
PRECISE
TITLE   TEST OF OPERATIONS ON THE HILBERT MATRIX I=6
SOLVE THE FOLLOWING SYSTEM OF 6 LINEAR EQUATIONS
D1.0,0.5,0.33333333333333333333,0.25,0.2,0.16666666666666666667
D2.45
D0.5,0.33333333333333333333,0.25,0.2,0.16666666666666666667
D0.142857142857142857143,1.592857142857142857143
D0.333333333333333333333,0.25,0.2,0.16666666666666666667
D0.14285714285714285714,0.125,1.21785714285714285714
D0.250,0.2,0.16666666666666666667,0.14285714285714285714
D0.1250,0.11111111111111111111,0.99563692063692063692
D0.20,0.16666666666666666667,0.14285714285714285714,0.125
D0.111111111111111111111,0.1,0.084563692063692063692
D0.166666666666666666667,0.14285714285714285714,0.125
D0.111111111111111111111,0.1,0.0909090909090909090909
D0.13376623376623376623
STOP
```

Fig. 3. Typical input for solving a set of linear equations.
The D in the first column of the data cards is required for double precision input.

```
MATRIX---N=  6
      1.00000000000000000000        0.50000000000000000000        0.333333333
      C.2000C000000000000000        0.16666666666666666667        2.450000000

      C.5000CC00000000000000        0.33333333333333333333        0.2500000000
      C.16666666666666666667        0.14285714285714285714        1.592857142

      0.33333333333333333333        0.25000000000000000000        0.2000000000
      0.14285714285714285714        0.12500000000000000000        1.217857142

      C.2500C000000000000000        0.20000000000000000000        0.166666666
      C.1250C000000000000000        0.11111111111111111111        0.995634920

      0.20000000000000000000        0.16666666666666666667        0.142857142
      C.11111111111111111111        0.10000000000000000000        0.845634920

      0.16666666666666666667        0.14285714285714285714        0.125000000
      0.10000000000000000000        C.09090909090909090909        0.736544011

    SOLLTIONS
      0.99999999999999997665        1.00000000000000065784        0.999999999
      C.99999999999998740932        1.00000000000000495135

        SOURCE PROGRAM LISTING

      PRECISE  TEST OF OPERATIONS ON THE HILBERT MATRIX  I=6
      TITLE    TEST OF OPERATIONS ON THE HILBERT MATRIX I=6
      FIXED  WITH 20 DECIMALS
      SOLVE THE FOLLOWING SYSTEM OF 6 LINEAR EQUATIONS
```

Fig. 4. A portion of the output of the solution of a set of six equations defined in Fig. 3. Note that the solutions should all be exactly 1. This loss of accuracy is characteristic of ill conditioned matrices and points up the need for carrying out calculations with higher precision. Results from a single precision run for n=6 would have been highly innaccurate.

```
PRECISE        TABLE FROM MTAC--V19--N92--OCT, 1965
GENERATE 1(1)5 ANC STO IN COL 1
HEACING CN 1 CARC
  660                                   CHIH-BING LING
FCRMAT ON 1 CARC
/39X,7CTABLE 3
PRINT TABLE HEAC
DCNTHEAD
DCNTSPACE
DCNTPAGE
FCRMAT ON 4 CARCS
86C ****************************************************************************
*****************/2C *,6X,1C*,35X,1C*,33X,1C*,6X,1C*/9C *     A *,15X,5CT
AN A,15X,1C*,14X,5CCOT A,14X,1C*,6X,1C*/86C ******************************
************************************************************
PRINT COLUMN HEACINGS
FCRMAT CN 1 CARC
2C *,0K1F0.5,2C *,5K1F26.34,2C *,5K1F23.31,3C  *,0K1F0.5,2C *
CCTD CF COL 1 TO CCL 3
TAND  1,2
SUBTRACT CCL 1 FROM 90. ANC STO IN COL 4
PRINT 1,2,3,4
GENERATE  6.(1.)45. AND STORE IN 1
SUBTRACT  1 FROM 90. AND STO IN  4
TAND  1,2
COTD  1  3
FCRMAT ON 1 CARC
2C *,0K1F0.5,2C *,5K1F25.32,03C  *,5K1F24.32,2C *,0K1F0.5,2C *
PRINT  1,2,3,  ANC 4
FORMAT ON 4 CARDS WHICH FOLLOW
86C ****************************************************************************
*****************/2C *,6X,1C*,35X,1C*,33X,1C*,6X,1C*/2C *,6X,1C*,15X,5CC
CT A,15X,1C*,14X,5CTAN A,14X,4C*  A,3X,1C*/86C ***************************
******************************************************************
PRINT TRAILING CCLUMN LABELS
```

Figure 5. Instruction for duplicating the trigonometric table of Figure 6. The two formats are required because of the reduction of the number of decimals tabulated above 6 degrees.

TABLE 3

$\theta°$	$\tan\theta$	$\cot\theta$	
1	0.01745 50649 28217 58576 51289 0	57.28996 16307 59424 68727 815	89
2	0.03492 07694 91747 73050 04026 3	28.63625 32829 15603 55075 651	88
3	0.05240 77792 83041 20403 88058 2	19.08113 66877 28211 06340 675	87
4	0.06992 68119 43510 41366 69210 6	14.30066 62567 11927 91012 805	86
5	0.08748 86635 25924 00522 20186 7	11.43005 23027 61343 06721 086	85
6	0.10510 42352 65676 46251 15024	9.51436 44542 22584 92968 3971	84
7	0.12278 45609 02904 59113 42311	8.14434 64279 74594 02382 5661	83
8	0.14054 08347 02391 44683 81177	7.11536 97223 84208 74823 0566	82
9	0.15838 44403 24536 29383 88831	6.31375 15146 75043 09897 9464	81
10	0.17632 69807 08464 97347 10904	5.67128 18196 17709 53099 4418	80
11	0.19438 03091 37718 48424 31942	5.14455 40159 70310 13472 3221	79
12	0.21255 65616 70022 12525 95917	4.70463 01094 78454 23358 6235	78
13	0.23086 81911 25563 11174 81456	4.33147 58742 84155 54554 6168	77
14	0.24932 80028 43180 69162 40399	4.01078 09335 35844 71634 5715	76
15	0.26794 91924 31122 70647 25537	3.73205 08075 68877 29352 7446	75
16	0.28674 53857 58807 94004 27581	3.48741 44438 40908 65069 6224	74
17	0.30573 06814 58660 35573 45420	3.27085 26184 84140 86530 8856	73
18	0.32491 96962 32906 32615 58714	3.07768 35371 75253 40257 0291	72
19	0.34432 76132 89665 24195 72658	2.90421 08776 75822 80257 9326	71
20	0.36397 02342 66202 36135 10479	2.74747 74194 54622 27876 1664	70
21	0.38386 40350 35415 79597 14484	2.60508 90646 93801 53625 8412	69
22	0.40402 62258 35156 81132 23481	2.47508 68534 16295 82524 0013	68
23	0.42447 48162 09604 74202 35321	2.35585 23658 23752 83393 9587	67
24	0.44522 86853 08536 16392 23670	2.24603 67739 04216 05416 3321	66
25	0.46630 76581 54998 59283 00062	2.14450 69205 09558 61635 6261	65
26	0.48773 25885 65861 42277 31111	2.05030 38415 79296 21689 9011	64
27	0.50952 54494 94428 81051 37069	1.96261 05055 05150 58230 4640	63
28	0.53170 94316 61478 74807 59159	1.88072 64653 46332 01236 0838	62
29	0.55430 90514 52768 91782 07631	1.80404 77552 71423 93738 1785	61
30	0.57735 02691 89625 76450 91488	1.73205 08075 68877 29352 7446	60
31	0.60086 06190 27560 41487 86644	1.66427 94823 50517 91103 0496	59
32	0.62486 93519 09327 50978 05108	1.60033 45290 41050 35532 6733	58
33	0.64940 75931 97510 57698 20629	1.53986 49638 14582 90482 6797	57
34	0.67450 85168 42426 63214 24609	1.48256 09685 12740 25478 7157	56
35	0.70020 75382 09709 77945 85227	1.42814 80067 42114 50216 0618	55
36	0.72654 25280 05360 88589 54668	1.37638 19204 71173 53820 7210	54
37	0.75355 40501 02794 15707 39564	1.32704 48216 20410 03715 9473	53
38	0.78128 56265 06717 39706 29500	1.27994 16321 93078 78031 1030	52
39	0.80978 40331 95007 14803 69914	1.23489 71565 35051 39855 6175	51
40	0.83909 96311 77280 01176 31273	1.19175 35925 94209 95870 5308	50
41	0.86928 67378 16226 66220 00956	1.15036 84072 21009 55587 6331	49
42	0.90040 40442 97839 94512 04772	1.11061 25148 29192 87014 3482	48
43	0.93251 50861 37661 70561 21856	1.07236 87100 24682 53294 6028	47
44	0.96568 87748 07074 04595 80273	1.03553 03137 90569 50695 8833	46
45	1.00000 00000 00000 00000 00000	1.00000 00000 00000 00000 0000	45
	$\cot\theta$	$\tan\theta$	$\theta°$

Values of the two coefficients σ_4 and σ_6 and the three functions

For the convenience of the reader, there is included in Table 3 a compilation of 25D values of the tangent and cotangent for arguments 1°(1°)89°. These data, which are required in computing the values of c, are here tabulated with the same range and precision as for the values of sine and cosine given by G. W. and R. M. Spenceley [4]. The only comparable table of decimal approximations to the tangent appears to be the relatively inaccessible 30D table of Herrmann [5].

Fig. 6. See the following figures for the degree to which this table layout is approximated via the instructions on the opposite page.

TABLE 3

A	TAN A	COT A	
1.	0.01745 50649 28217 58576 51289 0	57.28996 16307 59424 68727 815	89.
2.	0.03492 07694 91747 73050 04026 3	28.63625 32829 15603 55075 651	88.
3.	0.05240 77792 83041 20403 88058 2	19.08113 66877 28211 06340 675	87.
4.	0.06992 68119 43510 41366 69210 6	14.30066 62567 11927 91012 805	86.
5.	0.08748 86635 25924 00522 20186 7	11.43005 23027 61343 06721 086	85.
6.	0.10510 42352 65676 46251 15024	9.51436 44542 22584 92968 3971	84.
7.	0.12278 45609 02904 59113 42311	8.14434 64279 74594 02382 5661	83.
8.	0.14054 08347 02391 44683 81177	7.11536 97223 84208 74823 0566	82.
9.	0.15838 44403 24536 29383 88831	6.31375 15146 75043 09897 9464	81.
10.	0.17632 69807 08464 97347 10904	5.67128 18196 17709 53099 4418	80.
11.	0.19438 03091 37718 48424 31942	5.14455 40159 70310 13472 3221	79.
12.	0.21255 65616 70022 12525 95917	4.70463 01094 78454 23358 6235	78.
13.	0.23086 81911 25563 11174 81456	4.33147 58742 84155 54554 6168	77.
14.	0.24932 80028 43180 69162 40399	4.01078 09335 35844 71634 5715	76.
15.	0.26794 91924 31122 70647 25537	3.73205 08075 68877 29352 7446	75.
16.	0.28674 53857 58807 94004 27581	3.48741 44438 40908 65069 6224	74.
17.	0.30573 06814 58660 35573 45420	3.27085 26184 84140 86530 8856	73.
18.	0.32491 96962 32906 32615 58714	3.07768 35371 75253 40257 0291	72.
19.	0.34432 76132 89665 24195 72658	2.90421 08776 75822 80257 9326	71.
20.	0.36397 02342 66202 36135 10479	2.74747 74194 54622 27876 1664	70.
21.	0.38386 40350 35415 79597 14484	2.60508 90646 93801 53625 8412	69.
22.	0.40402 62258 35156 81132 23481	2.47508 68534 16295 82524 0013	68.
23.	0.42447 48162 09604 74202 35321	2.35585 23658 23752 83393 9587	67.
24.	0.44522 86853 08536 16392 23670	2.24603 67739 04216 05416 3321	66.
25.	0.46630 76581 54998 59283 00062	2.14450 69205 09558 61635 6261	65.
26.	0.48773 25885 65861 42277 31111	2.05030 38415 79296 21689 9011	64.
27.	0.50952 54494 94428 81051 37069	1.96261 05055 05150 58230 4640	63.
28.	0.53170 94316 61478 74807 59159	1.88072 64653 46332 01236 0838	62.
29.	0.55430 90514 52768 91782 07631	1.80404 77552 71423 93738 1785	61.
30.	0.57735 02691 89625 76450 91488	1.73205 08075 68877 29352 7446	60.
31.	0.60086 06190 27560 41487 86644	1.66427 94823 50517 91103 0496	59.
32.	0.62486 93519 09327 50978 05108	1.60033 45290 41050 35532 6733	58.
33.	0.64940 75931 97510 57698 20629	1.53986 49638 14582 90482 6797	57.
34.	0.67450 85168 42426 63214 24609	1.48256 09685 12740 25478 7157	56.
35.	0.70020 75382 09709 77945 85227	1.42814 80067 42114 50216 0618	55.
36.	0.72654 25280 05360 88589 54668	1.37638 19204 71173 53820 7210	54.
37.	0.75355 40501 02794 15707 39564	1.32704 48216 20410 03715 9473	53.
38.	0.78128 56265 06717 39706 29500	1.27994 16321 93078 78031 1030	52.
39.	0.80978 40331 95007 14803 69914	1.23489 71565 35051 39855 6175	51.
40.	0.83909 96311 77280 01176 31273	1.19175 35925 94209 95870 5308	50.
41.	0.86928 67378 16226 66220 00956	1.15036 84072 21009 55587 6331	49.
42.	0.90040 40442 97839 94512 04772	1.11061 25148 29192 87014 3482	48.
43.	0.93251 50861 37661 70561 21856	1.07236 87100 24682 53294 6028	47.
44.	0.96568 87748 07074 04595 80273	1.03553 03137 90569 50695 8833	46.
45.	1.00000 00000 00000 00000 00000	1.00000 00000 00000 00000 0000	45.
	COT A	TAN A	A

Figure 7. Results from the program shown in Figure 5.

```
PRECISE
TITLE ELEMENTARY AND TRANSCENDENTAL FUNCTIONS TO 25 FIGURES
FORMAT ON 1 CARD
1C 1F2.9,1P5K1S7(1E25.45
GENERATE -1.(.C5)1. IN COL 1
ARCSIN 1,2
HEADING ON 1 CARD
      X                          ARCSIN(X)
PRINT 1,2
ARCCOS 1,3
HEADING ON 1 CARD
      X                          ARCCOS(X)
PRINT 1,3
GENERATE .01(.01).1(.1)1.(1.)10.(10.)100.(100.)1000. IN COL 4
ARCTAN 4,5
HEADING ON 1 CARD
      X                          ARCTAN(X)
PRINT 4,5
ARCCOT 4,6
HEADING ON 1 CARD
      X                          ARCCOT(X)
PRINT 4,6
FIXED WITH 28
HEADING ON 1 CARD
          ARCSIN(X)
PRINT 2
HEADING ON 1 CARD
          ARCCOS(X)
PRINT 3
HEADING ON 1 CARD
          ARCTAN(X)
PRINT 5
HEADING ON 1 CARD
          ARCCOT(X)
PRINT 6
FLOATING WITH 28
HEADING ON 1 CARD
          ARCSIN(X)
PRINT 2
HEADING ON 1 CARD
          ARCCOS(X)
PRINT 3
```

Figure 8. A portion of a PRECISE program to compute tables of trigonometric functions. See Tables 8, 9, and 10 for some results of this calculation.

```
SOURCE PROGRAM LISTING

PRECISE     EXAMPLE OF A SMALL TABLE FOR LARGE ARGUMENTS
   GREATEST POWER OF TEN WILL BE LESS THAN 1000000
   FORMAT ON 1 CARD  WHICH FOLLOWS
1C 1F0.9,5K1P1S10(3E15.35 FIRST NON-BCD FIELD BLANK TERMINATES FORMAT
   HEADING ON 2 CARDS WHICH FOLLOW
         X                                         EXPF(X)
EXPF(-X)                                         LOGF(X)
GENERATE 1.(1.)10.(10.)1E2(1E2)1.E3(1E3)1.0E4(1E4)1E5(5E5)1E6 IN COL 1
   EXP OF COLUMN 1 AND STORE IN COLUMN 2
   DIVIDE 1. BY COLUMN 2 AND STORE IN COLUMN 3
   LOGE OF COLUMN 2 AND STORE IN COLUMN 4
    PRINT COLUMNS 1, 2, 3, AND  4

END OF SOURCE PROGRAM LISTING.

    71 LINES OUTPUT.
```

Figure 9. A program listing showing the PRECISE instructions for calculating the exponential function for positive and negative arguments. See Figure 10 for the result of this calculation. The results are printed to only 15 decimal places.

PAGE 1 EXAMPLE OF A SMALL TABLE FOR LARGE ARGUMENTS

X	EXPF(X)	EXPF(-X)	LOGF(X)
1.	(0) 2.71828 18284 59045	(-1) 3.67879 44117 14423	(0) 1.00000 00000 00000
2.	(0) 7.38905 60989 30650	(-1) 1.35335 28323 66127	(0) 2.00000 00000 00000
3.	(1) 2.00855 36923 18767	(-2) 4.97870 68367 86394	(0) 3.00000 00000 00000
4.	(1) 5.45981 50033 14424	(-2) 1.83156 38888 73418	(0) 4.00000 00000 00000
5.	(2) 1.48413 15910 25766	(-3) 6.73794 69990 85467	(0) 5.00000 00000 00000
6.	(2) 4.03428 79349 27351	(-3) 2.47875 21766 66358	(0) 6.00000 00000 00000
7.	(3) 1.09663 31584 28459	(-4) 9.11881 96555 45162	(0) 7.00000 00000 00000
8.	(3) 2.98095 79870 41728	(-4) 3.35462 62790 25118	(0) 8.00000 00000 00000
9.	(3) 8.10308 39275 75384	(-4) 1.23409 80408 66795	(0) 9.00000 00000 00000
10.	(4) 2.20264 65794 80672	(-5) 4.53999 29762 48485	(1) 1.00000 00000 00000
20.	(8) 4.85165 19540 97903	(-9) 2.06115 36224 38558	(1) 2.00000 00000 00000
30.	(13) 1.06864 74581 52446	(-14) 9.35762 29688 40175	(1) 3.00000 00000 00000
40.	(17) 2.35385 26683 70200	(-18) 4.24835 42552 91589	(1) 4.00000 00000 00000
50.	(21) 5.18470 55285 87072	(-22) 1.92874 98479 63918	(1) 5.00000 00000 00000
60.	(26) 1.14200 73898 15684	(-27) 8.75651 07626 96520	(1) 6.00000 00000 00000
70.	(30) 2.51543 86709 19167	(-31) 3.97544 97359 08647	(1) 7.00000 00000 00000
80.	(34) 5.54062 23843 93510	(-35) 1.80485 13878 45415	(1) 8.00000 00000 00000
90.	(39) 1.22040 32943 17841	(-40) 8.19401 26239 90515	(1) 9.00000 00000 00000
100.	(43) 2.68811 71418 16135	(-44) 3.72007 59760 20836	(2) 1.00000 00000 00000
200.	(86) 7.22597 37681 25749	(-87) 1.38389 65267 36738	(2) 2.00000 00000 00000
300.	(130) 1.94242 63952 41256	(-131) 5.14820 02224 12014	(2) 3.00000 00000 00000
400.	(173) 5.22146 96897 64144	(-174) 1.91516 95967 14006	(2) 4.00000 00000 00000
500.	(217) 1.40359 22178 52837	(-218) 7.12457 64067 41286	(2) 5.00000 00000 00000
600.	(260) 3.77302 03009 29940	(-261) 2.65039 65530 04311	(2) 6.00000 00000 00000
700.	(304) 1.01423 20547 35005	(-305) 9.85967 65437 59771	(2) 7.00000 00000 00000
800.	(347) 2.72637 45721 12567	(-348) 3.66787 45841 77687	(2) 8.00000 00000 00000
900.	(390) 7.32881 42223 07422	(-391) 1.36447 72123 65683	(2) 9.00000 00000 00000
1000.	(434) 1.97007 11140 17047	(-435) 5.07595 88975 49457	(3) 1.00000 00000 00000
2000.	(868) 3.88118 01942 84369	(-869) 2.57653 58729 61150	(3) 2.00000 00000 00000
3000.	(1302) 7.64620 09890 54705	(-1303) 1.30783 90189 21250	(3) 3.00000 00000 00000
4000.	(1737) 1.50635 59700 50525	(-1738) 6.63853 71046 55673	(3) 4.00000 00000 00000
5000.	(2171) 2.96762 83840 23667	(-2172) 3.36969 41483 08918	(3) 5.00000 00000 00000
6000.	(2605) 5.84643 89565 02115	(-2606) 1.71044 28994 12899	(3) 6.00000 00000 00000
7000.	(3040) 1.15179 00508 06878	(-3041) 8.68213 78540 25194	(3) 7.00000 00000 00000
8000.	(3474) 2.26910 83085 06858	(-3475) 4.40701 74889 89013	(3) 8.00000 00000 00000
9000.	(3908) 4.47030 47331 65443	(-3909) 2.23698 39634 88985	(3) 9.00000 00000 00000
10000.	(4342) 8.80681 82256 62922	(-4343) 1.13548 38653 14736	(4) 1.00000 00000 00000
20000.	(8685) 7.75600 47259 86861	(-8686) 1.28932 36083 90094	(4) 2.00000 00000 00000
30000.	(13028) 6.83057 23779 14884	(-13029) 1.46400 61544 96327	(4) 3.00000 00000 00000
40000.	(17371) 6.01556 09309 53052	(-17372) 1.66235 53671 52052	(4) 4.00000 00000 00000
50000.	(21714) 5.29779 51644 30315	(-21715) 1.88757 76978 20509	(4) 5.00000 00000 00000
60000.	(26057) 4.66567 19009 93380	(-26058) 2.14331 40204 03123	(4) 6.00000 00000 00000
70000.	(30400) 4.10897 24332 63187	(-30401) 2.43369 84884 70605	(4) 7.00000 00000 00000
80000.	(34743) 3.61869 73314 00875	(-34744) 2.7[illegible]42 53666 99233	(4) 8.00000 00000 00000
90000.	(39086) 3.18692 09611 33901	(-39087) 3.13782 49168 88437	(4) 9.00000 00000 00000
100000.	(43429) 2.80666 33604 26123	(-43430) 3.56294 95653 09373	(5) 1.00000 00000 00000
600000.	(260576) 4.88812 10366 02819	(-260577) 2.04577 58564 32121	(5) 6.00000 00000 00000
1000000.	(434294) 3.03321 53968 02088	(-434295) 3.29683 14780 88559	(6) 1.00000 00000 00000

Figure 10. Results of the PRECISE calculation outlined in Figure 9.

X	n	$COS(X) \cdot 10^{-n}$
0.01	(-1)	9.99950 00041 66652 77780 25793
0.02	(-1)	9.99800 00666 65777 78412 69559
0.03	(-1)	9.99550 03374 89875 16272 15871
0.04	(-1)	9.99200 10666 09779 40314 57076
0.05	(-1)	9.98750 26039 49662 46562 87081
0.06	(-1)	9.98200 53993 52041 65547 66169
0.07	(-1)	9.97551 00025 32795 74620 90839
0.08	(-1)	9.96801 70630 26193 84977 70677
0.09	(-1)	9.95952 73301 19942 53092 83937
0.10	(-1)	9.95004 16527 80257 66095 56199
0.20	(-1)	9.80066 57784 12416 31124 19652
0.30	(-1)	9.55336 48912 56060 19642 31023
0.40	(-1)	9.21060 99400 28850 82798 52673
0.50	(-1)	8.77582 56189 03727 16116 28158
0.60	(-1)	8.25335 61490 96782 97240 95250
0.70	(-1)	7.64842 18728 44884 26255 85999
0.80	(-1)	6.96706 70934 71654 20920 74998
0.90	(-1)	6.21609 96827 06644 56484 71615
1.00	(-1)	5.40302 30586 81397 17400 93661
2.00	(-1)	-4.16146 83654 71423 86997 56823
3.00	(-1)	-9.89992 49660 04454 57271 57279
4.00	(-1)	-6.53643 62086 36119 14639 16818
5.00	(-1)	2.83662 18546 32262 64466 63917
6.00	(-1)	9.60170 28665 03660 20545 65230
7.00	(-1)	7.53902 25434 33046 38141 19752
8.00	(-1)	-1.45500 03380 86135 25868 84138
9.00	(-1)	-9.11130 26188 46769 88368 29471
10.00	(-1)	-8.39071 52907 64524 52258 86395
20.00	(-1)	4.08082 06181 33919 86062 26786
30.00	(-1)	1.54251 44988 75840 50718 66215
40.00	(-1)	-6.66938 06165 22618 44384 09278
50.00	(-1)	9.64966 02849 21132 74068 95706
60.00	(-1)	-9.52412 98041 51562 92693 81660
70.00	(-1)	6.33319 20308 62998 32332 01150
80.00	(-1)	-1.10387 24383 90475 58117 86666
90.00	(-1)	-4.48073 61612 91701 52365 47732
100.00	(-1)	8.62318 87228 76839 34101 93851
200.00	(-1)	4.87187 67500 70059 10354 74790
300.00	(-2)	-2.20966 19278 68394 26890 75598
400.00	(-1)	-5.25296 33864 25359 77291 94961

Figure 11. A portion of a table of trigonometric function computed on PRECISE. The values were checked against existing tables up to the snaded area.

```
    SOURCE PROGRAM LISTING

  PRECISE      EXAMPLE OF TAPE OPERATIONS
   GENERATE        1.(1.)40., STORE IN COLUMN1
   ADD             1,1,2
   SORT            1,3
   PRINT           1,2,3,.5
   REWIND TAPES 3,4,AND 2
   PUT ON TAPE 2, COLUMNS 1,2,3
   POSITION TAPE 2 BACK 1 FILE, 3 RECORDS,  0,0 FORWARD
   GET FROM TAPE 2, COLUMNS 5,6,7
   PRINT COLUMNS 5,6,7
   POSITION TAPE 2,0,0,1 FILE FORWARD,0
   PUT ON TAPE 2 EVERYTHING
   TRANSFER NEXT STATEMENTS TO TAPE 3
   ENDTRA
   ENDFILE TAPE  3
   REWIND TAPE 3
   SWITCH INPUT TO TAPE 3
   ERASE COLUMNS 1,2,3,4,5,6,7,8                          (TAPE THREE)
   POSITION TP 2 1 FILE, AND1 RECORD BACKWARD, 0,0 FORWARD(TAPE THREE)
   GET FROM TAPE 2 EVERYTHING                             (TAPE THREE)
   PRINT  1,2,3,4                                         (TAPE THREE)
   PRINT  5,6,7,8                                         (TAPE THREE)
   REWIND TAPE 2                                          (TAPE THREE)
   SWITCH INPUT BACK TO REGULAR INPUT                     (TAPE THREE)
   PRINT  COLUMN 1
  STOP
```

Figure 12. Some examples of the tape manipulation features in PRECISE.

APPENDIX I - JOB DEFINITION OF PRECISE PROGRAMS

PRECISE was written to operate as a sub-monitor under the IBSYS or DC-IBSYS Monitor System for the IBM 7090/7094, or as an IBMAP program to be run under the IBJOB monitor under IBSYS. Either of the two versions may be produced by setting one parameter and making an absolute or relocatable assembly of the source program via the IBMAP Assembler of IBJOB.

JOB Definition when PRECISE is a Monitor

A PRECISE job will consist at most of the following cards:

```
$JOB
$EXECUTE          PRECISE
$ID
   [ SOURCE 1 ]
       .
   [ SOURCE n ]
STOP
end of file
```

Where [SOURCE i] consists of the PRECISE command followed by as many as desired of the commands in the PRECISE language (excluding the PRECISE and STOP commands).

The requirements as to need and format of the $JOB and $ID cards will vary among installations.

JOB Definition with PRECISE under IBJOB

A PRECISE job when using the relocatable deck under the IBJOB Monitor will be made up as follows.

```
$JOB
$EXECUTEΔΔΔΔΔΔΔIBJOB
$ID
$IBJOB
  [ PRECISE program decks ]
$DATA
  [ SOURCE 1 ]
      .
  [ SOURCE n ]
STOP
end of file
```

Where [SOURCE i] consists of the PRECISE command followed by as many as desired of the commands in the PRECISE language (excluding the PRECISE and STOP commands).

APPENDIX II - FORMAT DESCRIPTION

A format consists of any number of field specifications. The general field specification is of the following form.

aSb(c)gPnTd•wZ

a,b,c,g,n,d, and w are positive integers.

T is the type conversion character and T is I, E, F, O, A, or X.

Z is a slash, comma, or blank, and Z serves as a field specification separator.

Each non-numeric Hollerith character in a field specification indicates what is to be done with the integer just preceding it.

Each field specification is executed n times and n data fields are printed, each having a total column width w with the spread of information being determined by a,b,c,g, and d.

T = I specifies full word binary to decimal integer conversion.

T = E specifies binary to floating point decimal conversion of multiple precision numbers.

T = F specifies binary to fixed point decimal conversion of multiple precision numbers.

T = O specifies full word binary to octal conversion.

T = A specifies BCD to BCD conversion.

T = X specifies insertion of blank characters.

If either of the integers a,b,c, or g is zero then that integer and the following Hollerith character may be omitted.

The format is scanned from left to right. Conversion as indicated by a specification is completed before checking on the separator Z. If Z is a blank, then all information converted at this point is printed and if there is still more information to be converted and printed, scanning goes back to the beginning of the format. If Z is a slash, all information converted at this point is printed and scanning for the next line starts immediately after the slash. K consecutive slashes separating two specifications cause K-1 blank lines to be printed. If Z is a comma, conversion continues with the specification following the comma being scanned.

Binary to Decimal Integer Conversion (T=I) {gPnIwZ}

n single celled binary integers are converted and printed as decimal integers, each with a total column width w; and each number is enclosed in parenthesis if $g \neq 0$. The restrictions are $n>0$, $w \leq 26$.

Example: Suppose the binary equivalent of the integers 1, -2, 3, 7000, and -56789 are to be printed. Then the format 1P1I4,4I5 would cause the following line to be printed.

```
 (1)   -2    3 700056789
```

Note that the negative sign of the last number would not be printed since w = 5.

Binary to Floating Point Decimal Conversion (T=E) {aSb(c)gPnEd•wZ}

n 3 cell multiple precision numbers are converted and printed, each number having a total column width w. Information within the column is arranged as follows from left to right.

1. w-a-b-c-d-2 blanks.

2. the decimal exponent of the number occupying b places and in parenthesis if $g \neq 0$.

3. The sign of the number (blank for +, or a minus sign).

4. a significant digits of the number.

5. a decimal point.

6. d significant decimal digits (rounded).

7. the decimal exponent of the number occupying c places and in parenthesis if $g \neq 0$.

Restrictions are $n>0, w \geq a+b+c+d+2$, $b \leq 26$, $c \leq 26$, $d+a<30$.

Example: The format 1S4(1P1E5.13,3)1E10.19 would cause the binary equivalent of 12345.67 and -.00765432 to be printed as follows.

```
(4) 1.23457    -.7654320000 -2
```

Binary to Fixed Point Decimal Conversion (T=F) {aSnFd•wZ}

n 3 cell multiple precision numbers are converted and printed as fixed point decimal numbers, each number having a total column width w. Information within the column is arranged from left to right as follows. Before conversion, the number is multiplied by 10 to the a power.

1. The integral part of the number (with sign) occupying w-d-1 spaces.

2. A decimal point.

3. d decimal places (rounded) of the fractional part of the number.

Restrictions are n>0, w≥d+2+number of digits in integral part.

This subroutine arbitrarily refuses to print fixed point multiple precision numbers whose absolute value is greater than $2^{35}-1$. If such a number is found, the specification is treated as E type rather than F type and the number is printed in floating point form, but the power of ten exponent cannot be printed unless either b (or c) is written in the specification. b and c are ignored in F type conversion, so no harm is done by specifying one of the two just in case large numbers are printed.

Example: The format 3)3F10.20 would cause the binary equivalent of -101.01, 10^{40}, and .00004778966 to be printed as follows.

```
-101.0100000000      .1000000000 41        .0000477897
```

Binary to Octal Conversion (T=O) {aSb(c)gPnOd•wZ}

n* single celled binary integers are converted and printed as octal integers, with locations if desired. Each number has a total column width w and information within the column is arranged from left to right as follows.

1. w-a-b-c-d blanks.

2. The location as a decimal integer occupying b places and in parenthesis if g = 1, 3, 9, or 11.

3. The location as an octal integer occupying c places and in parenthesis if g = 2, 3, 10, or 11.

4. a blanks.

5. The last d octal digits of the number.

Restrictions are d≤12, b≤26, c≤26, w≥a+b+c+d, n>0. If g≥8, then as far as the calling sequence is concerned, the number described above is not counted as having been printed; thus allowing a single cell to be printed both in octal and decimal with or without decimal and/or octal locations; or if d = 0 allowing a location to be printed with either a single celled number or multiple precision number.

Example: The format 1S5(9P1o0.7,1I6,2S3)1o12.21,1o3.7 would cause the octal numbers -17,314631463146 and 7777777 in decimal locations 100, 101, and 102 to be printed as follows.

(100) -15 145 314631463146 777

* If n>1 then g must be less than 8.

BCD Conversion (T=A) {nAwZ}

n BCD words are printed as n BCD fields with each field having a total column width w. If w>6, then each field consists of w-6 blanks followed by the BCD word consisting of 6 characters. If w≤6 then the field consists of the leftmost w characters of the BCD word.

Blank Insertion (T=X) {nXZ}

n blanks are inserted into the print line.

BCD Characters Within the Format

All of the previously described field specifications may be preceded by iC followed by i Hollerith characters. The i characters will be printed just before the first number printed by the specification which follows.

Printing of Tables {mKaSb(c)gPnTd•zW}

To make numbers with many digits more readable, an additional feature is provided in the format specification for E and F type conversion. Either the F or the E specification as described previously may be preceded by mK.

The effect of mK is to cause blocks of m digits to the right of the decimal point to be separated by one blank. The separation will occur on both E and F type, and it will continue until reset by another mK appearing in some specification.

When this feature is used, the column width w must be increased enough to allow j extra spaces.

Let $d/m=k+R$, k an integer and $0 \le R < m$.

$$\text{Then } j = \begin{cases} 0 \text{ if } m=0. \\ k \text{ if } R>0. \\ k-1 \text{ if } R=0. \end{cases}$$

The PRECISE system consists of about 8000 IBMAP instructions in the form of a PREST deck, which is about 3/4 of a box of cards. To assemble PRECISE as an absolute program (which can be edited into IBSYS) use the ALTER feature of IBJOB to replace card 2 with the IBMAP symbolic instruction:

RELMODΔSETΔΔΔΔΔ2

and use the ABSMOD option on the $IBMAP card. To make PRECISE a sub-monitor under IBSYS, enter "PRECIS" into the name table of IBSYS, and then insert the absolute program deck into the desired position of the system tape.

A small dummy IBMAP program is included with the system and this program should be discarded if making an absolute assembly.

To assemble PRECISE as a relocatable program, use the ALTER feature of IBJOB to replace card 2 with the IBMAP symbolic instruction:

RELMODΔSETΔΔΔΔΔ1

and include the small dummy IBMAP program to be assembled. This dummy program has the deck name USERS and has the entries USERSA, USERSB, USERSC, USERSD, and USERSE. This program is necessary because a relocatable assembly of the system generates calls to the above entries. These external calls are automatically deleted from an absolute assembly. The USERS program also has a common block definition of key calls of the system. A complete listing of USERS is given later. The relocatable assembly of PRECISE also defines many entry points to various parts of the system. By use of the entries and COMMON values, new commands may be implemented in the system when the code to accomplish the command function is written and made part of USERS program. Ways of adding commands are given later.

The PRECISE system consists of 5 blocks of coding.

1. The system control part of the system.
2. UOMΔMPP package, modified for IBMAP and I/O calls to UOMΔIOS (3).
3. UOMΔIOS package by A. Beam and available as SHARE distribution number 3444.
4. The PRECISE coding.
5. The vocabulary of commands.

Available storage is computed by using .JCOR in relocatable versions and SYSCOR in absolute versions.

The entry points in relocatable versions are

1. All entries defined in the UOMΔMPP write-up.
2. All entries defined in the UOMΔIOS write-up.
3. INVLD which may be called when an invalid command is detected by the user's program.

The Vocabulary

Each legal command defined to the system must have a two word entry in the vocabulary. Only two types of entries are described here, and should be sufficient for expansion of the system. The entry

```
BCIΔΔΔΔ1, name
TXLΔΔΔΔcode,4,0
```

is used when the coding at location code is part of the main system.

The entry

```
BCIΔΔΔΔ1, name
CALLΔΔΔcode
ORGΔΔΔΔ*-2
```

is used when the coding at location code is external to the system. Name is the PRECISE command word and consists of 6 or less alphabetic characters, right justified with leading zeros. For example, the command word SQRT appears in the first entry type above as BCI 1,00SQRTΔ. The coding at location code must be a subroutine which is entered via a TSX code, 4 and which exits via a TRAΔ1, 4Δ. The coding utilizes the COMMON block, and entry points of PRECISE to carry out the desired function. Entries to the vocabulary must be in ascending sort order on the BCI command name words.

The COMMON block

A listing of the COMMON block appears in the listing of USERS program. To explain the most used values in the block, suppose the following PRECISE command has been received by the system.

name C(1), C(2),..., C(n)

Where name is the command word and C(i), i=1,...,n are arguments to the command.

When control is given to the code for execution of the command, the following information is available in the COMMON block.

S : The address portion of S contains the matrix origin.

NC : The address portion of NC contains the maximum column number available.

NR : The address portion of NR contains the maximum row number available. Hence the current size of the matrix is the product of NR by NC.

ARGS : The origin of a vector containing the arguments C(i) of the command, 3 words per argument. Argument C(i) will be in locations ARGS+3(i-1)+j, j=0,1,2; if C(i) is a multiple precision number; and in location ARGS+3i-1 if C(i) is an integer.

STATS : The origin of a vector containing one word per C(i). The word at STATS+i-1 will be 0 if argument C(i) is an integer, and will be non-zero if argument C(i) is a multiple precision number.

PICKUP : The origin of a vector containing one word per C(i). Each word at PICKUP+i-1 is of the form

PZE x,I,A

where x may be any value.
I will be 0 if C(i) is a multiple precision number.
I will be 1 if C(i) is an integer.
A is the origin of the vector C(i) if I=1 or the origin of a multiple precision number if I=0.

NARGS : The address portion of NARGS contains n= the number of arguments on the PRECISE command currently under consideration.

NRA : The address portion of NRA is the current operating length of columns of the matrix. Operations on rows should be carried out on the 1st through NRA th row.

BUFFER : The PRECISE command (input card image) currently under consideration is stored in BUFFER+I, I=1,2,...,12.

EXTRAS : The origin of a 100 word vector which may be used as temporary storage.

The above information is sufficient for a subroutine to operate on one or more columns in any desired fashion. Columns should be operated upon only through the NRA th row of the matrix.

Expanding the System

It may be desirable to add commands to the PRECISE language. Coding for new commands should probably be checked out using the relocatable IBJOB version of PRECISE.

The easiest way to add and check out a new command is to use one of the entries in the USERS program. Then at a later time the permanent vocabulary entry and the coding can be made a part of the system.

A listing of the USERS program follows, and includes the COMMON block which is necessary for any new relocatable programs which may be added to the system.

```
$IBMAP USERS    ()OK
* USERS PROGRAM TO ADD AND CHECK FUNCTIONS
        ENTRY   USERSA
        ENTRY   USERSB
        ENTRY   USERSC
        ENTRY   USERSD
        ENTRY   USERSE
*
* C O M M O N   B L O C K
*
        USE     //
S       BSS     1  ADDRESS DEFINES MATRIX ORIGIN
NC      BSS     1  MAXIMUM NUMBER OF COLUMNS AVAILABLE
NMAX    BSS     1  NUMBER OF CELLS IN MATRIX
NR      BSS     1  MAXIMUM NUMBER OF ROWS IN MATRIX
NRA     BSS     1  COUNT OF ROWS READ IN
SKIPOP  BSS     1  0 UNTIL AN ID IS FOUND
EXTRAS  BSS     100 USED TO STORE SUMS---AVAILABLE FOR TEMP. USE
SUMSWT  BSS     1  NOT 0 IF COLUMN SUMS WANTED
TYPECH  BSS     1  TYPE OF PRINT=0,1,...,10
BLOCKC  BSS     1  PRINT BLOCK COUNT---10 LINE/BLOCK---5 BLOCKS/PAGE
ERSKIP  BSS     1  SET NOT 0 WHEN BAD OP ENCOUNTERED
INVEC   BSS     1  SET NOT 0 BY 'READ' COMMAND
NERROR  BSS     1  ERROR TYPE IN DECREMENT = 0,1,2,...
LINENO  BSS     1  LINE COUNT FOR PRINTING
PAGENO  BSS     1  PAGE COUNT FOR PRINTING
PICKUP  BSS     36 PZE X,I,A---I=0 IF CONSTANT,ELSE 1---A=ARG ADDRESS
ARGS    BSS     3*36 START OF ARGUMENT VECTOR---INTEGERS ARE IN 3RD CELL
STATS   BSS     36 STATS VECTOR---INTEGER IF 0,ELSE MPP NUMBER
BUFFER  BSS     21 INPUT CARD IMAGE IN BUFFER+I, I=1,2,...
NARGS   BSS     1  NUMBER OF ARGUMENTS ON CARD
ENDMAT  BSS     1 ADDRESS DEFINES LAST CELL AVAILABLE FOR MATRIX
        CONTRL  //
        USE     PREVIOUS
*
* END OF COMMON BLOCK
* USE SYMBOLS J,H(LESS THAN 6),K(EXCEPT K),Q,V,X,Y(ALL)
USERSA  NULL    *
USERSB  NULL    *
USERSC  NULL    *
USERSD  NULL    *
USERSE  NULL    *
        TRA     1,4
        END
```

Note that all the entries in the USERS program do nothing except return to the main PRECISE package.

Hence, one could write the commands

USERSA

.

.

.

USERSE

with any arguments desired. The commands would be executed, but nothing useful would be accomplished. However, on entry to the USERS program certain information would have already been set up in the COMMON block; namely the PICKUP, ARGS, and STATS vectors which define the arguments on the command and NARGS which gives the number of arguments. This information along with NRA (which describes the current number of rows in the matrix) is sufficient to operate on the arguments which appear on the command.

For example, suppose we need a PRECISE command which computes the product of all the numbers in the column specified by the first argument and stores this product in the column specified by the second argument of the command. For check out we could call the command name USERSA and by writing the command

USERSA COL 5,4

the product of all values in column 5 will be stored over all values in column 4. The coding listed below will produce the desired result if the entire block of coding is inserted into the USERS program (listed above) between the two cards labeled USERSA and USERSB, and the USERS program is re-assembled.

```
* USERSA C1,C2 --- INPUT C1 AND OUTPUT IN C2
* COMPUTE A(1)*A(2)*...*A(NRMAX) AND STORE IN SPECIFIED VECTOR
PRODUC NULL
       SXA    UA5,4
       AXT    0,1
       LXA    NRA,2
       CLA    PICKUP
       STD    UA2
       CLA    PICKUP+1
       STD    UA4
       TSX    ENTRY,4
              CLA1,0,UA6
              DPFLTI
              DPEXIT
UA1    TSX    ENTRY,4
UA2           MPY,1,**
              DPEXIT
       TXI    *+1,1,-3
       TIX    UA1,2,1
       AXT    0,1
       LXA    NRA,2
UA3    TSX    ENTRY,4
UA4           STO,1,**
              DPEXIT
       TXI    *+1,1,-3
       TIX    UA3,2,1
UA5    AXT    **,4
       TRA    1,4
UA6    PZE    1
```

If, at some later,time, it is desired to make the product command a permanent part of the system, then a vocabulary entry (for example, PRODUC) could be made and the above coding could be inserted into the PRECISE system. After a re-assembly, the command

PRODUCT OF C(1) TO C(2)

would be a part of the PRECISE language. The vocabulary entry would consist of the two cards.

BCIΔΔΔΔΔ1,PRODUCT
TXLΔΔΔΔΔPRODUC

which would be inserted between the entries for PRNT10 and RAISEI.

The above method of adding a command will allow the command PRODUCT to work in both the relocatable and absolute versions of the system.

APPENDIX IV

MULTIPLE PRECISION PACKAGE

(UOM MPP)

for the

IBM 7090/7094

by

Alfred E. Beam

Part of the work on this package was done while the author was employed at the National Bureau of Standards. The final version of the package and its incorporation under IBSYS was performed at the Computer Science Center of the University of Maryland and was supported by grant NsG-398 of the National Aeronautics and Space Administration.

Introduction

Many scientific problems lead to computational requirements involving extensive accuracy or extrememly large data ranges. This report describes an arithmetic program package for the IBM 7090 or IBM 7094 computer which essentially extends the standard word length and the normal data range for these machines. More precisely the package interprets and executes a set of pseudo-instructions in a multiple precision mode and provides input-output for several types of data, in particular multiple precision numbers. Pseudo-instructions are provided not only for the standard arithmetic operations but also for most of the elementary functions. In addition certain subroutines have been provided for solving a large class of problems requiring multiple precision computations.

Numbers operated upon are in a floating point form. Internally a number occupies 3 machine cells; the fractional part of the number is stored in the first two cells and in the first part of the third cell. The power-of-two exponent is stored in the remaining part of the third cell. The normal format of the third cell specifies that 26 bits are for the fractional part of the number and the last 9 bits are for the power-of-two exponent. A subroutine has been included in the package which makes it possible to reset the format of the third cell and thus vary the possible range of the data.

In brief, the package will effectively transform a 7090 into a machine which

1. is about 3 1/2 times as accurate.

2. is about 20 times slower.

3. has about 1/3 the storage capacity.

4. has the capability of handling numbers whose magnitudes are very large or very small. For example, the package can compute e to the millionth power correct to 22 significant decimal digits.

This package has already found wide acceptance and has been used extensively at the University of Maryland, the NASA-Goddard Space Flight Center, the National Bureau of Standards, Washington, D.C., and the General Motors Corporation, Detroit, Michigan. Its main uses have been for computing tables, checking of double precision IBM 7090 codes and for calculation of rational approximation coefficients.

Identification

a. 7090/7094 Multiple Precision Package UOM MPP
b. A. Beam, September 23, 1963
c. Computer Science Center, University of Maryland, College Park, Maryland

Purpose

To interpret and execute certain pseudo-instructions in a multiple precision mode, and to provide input-output for several types of data including multiple precision numbers. Also to provide a few subroutines so that the package can be used for solving a large class of problems which require high precision and or computation using numbers of very great or very small magnitudes.

Restrictions

The symbolic deck for MPP contains many symbols and is set up as a relocatable FAP subroutine with many entries. In this form it is usable as a subprogram by programs which run under the FORTRAN II Monitor under IBSYS. All I/O is self-contained except for the use of IOEX, hence the package could be used under any system which operates under IBSYS.

Underflow-overflow triggers and indicators are not saved by the package.

Method

In the interpretive mode successive pseudo-instructions are brought into the accumulator and control is sent to the proper place within the package for execution. Numbers operated on, and results given, while in the interpretive mode are of the form

$$N=F\cdot 2^{P},\ 1/2 \leq |F| < 1 \text{ or } F=0, \text{ and } -2^{b-1} \leq P < 2^{b-1}, 35 > b \geq 9.$$

N is stored in 3 consecutive cells with the sign of F in each cell. There are b (at present b=9, but this may be changed by use of a subroutine to be described later) bits at the end of the third cell which contain $P+2^{b-1}$ and the 105-b bits (not including signs) of the 3 cells contain $|F|$. Hence "multiple precision number" in this write-up is taken to mean a number which has at least 21 but no more than 28.8 significant decimal digits, and the potential range of a number is increased as the number of significant digits is decreased.

Arithmetic operations are coded using fixed point machine instructions and all yield a (105-b) bit rounded result. On underflow the result is automatically set to zero. If overflow is detected, a message is printed and execution is normally terminated. A subroutine is described later which allows the user to get control if overflow is detected.

The method used in the elementary functions is described later.

Usage

Usage of subroutines is described later. The interpretive mode is entered by giving the instruction

```
L      TSX     ENTRY,4
```

and location L is followed by as many as desired of the pseudo-instructions. Interpretation begins at location L+1 and continues until one of several pseudo-instructions is found which cause departure from the interpretive mode.

Storage

MPP requires about $(4000)_{10}$ storage locations. COMMON storage is not used.

Timing

Timing for a code using MPP should be from 20 to 25 times that for the same code using single precision floating point arithmetic.

All of the elementary functions (except the square root and integer exponentiation) in the package are computed by series, and computation proceeds until there is no contribution. Hence timing for the elementary functions will decrease somewhat as b is increased.

Checkout

Several versions of MPP have been used for a total of many hours on IBM 704's, 7090's, and 7094's.

Description of the Pseudo-Instructions

1. P.I. will be used to mean a pseudo-instruction interpreted by this package.

2. M.P.N. will be used to mean "multiple precision number."

3. The address of the M.P.N. in cells A, A + 1, A + 2 shall be A, and (A) specifies the M.P.N. at address A.

4. A three cell pseudo-accumulator will be designated by ACC and (ACC) refers to the M.P.N. in the ACC.

5. ACC1 will refer to the first cell of the ACC.

6. B will indicate a single location and (B) may be any 36 bit configuration.

7. $(ACC1)_f$ means (ACC1) treated as a machine floating point number.

8. $(ACC1)_i$ means (ACC1) treated as an integer i and $o \leq |i| < 2^{35}$.

9. $(j) \leftarrow (k)$ means that the contents of j are replaced by the contents of k.

10. Each P.I. is written in the following manner:

    i) The address (bits 21-35) part of the P.I. contains the pseudo-operation.

    ii) The tag (bits 18-20) part of the P.I. contains a tag of 0, 1, or 2. 3, 5, 6, and 7 may be used on the 7094.

    iii) The decrement (bits 3-17) part of the P.I. contains the address.

    iv) The prefix (sign, 1, and 2 bits) is always zero.

11. If a P.I. contains I in the tag position, that P.I. is indexable with either a zero, one, or two. Index 4 must not be used while in the interpretive mode. 3, 5, 6, and 7 may be used on the 7094.

12. If nothing is written in tag and decrement parts of a P.I., then that part of the P.I. may be utilized by the programmer.

List of the Pseudo-Instructions

In the following list, the general form of each P.I. is followed by its function and restrictions. If not specifically mentioned, the next P.I. to be interpreted is the next one in sequence.

1. CLS,I,A : $(ACC) \leftarrow -(A)$.
2. CLA,I,A : $(ACC) \leftarrow (A)$.
3. STO,I,A : $(A) \leftarrow (ACC)$.
4. ADD,I,A : $(ACC) \leftarrow (ACC)+(A)$.
5. SUB,I,A : $(ACC) \leftarrow (ACC)-(A)$.
6. MPY,I,A : $(ACC) \leftarrow (ACC)\cdot(A)$.
7. DVH,I,A : $(ACC) \leftarrow (ACC)/(A)$, error message if $(A)=o$.
8. CHS : $(ACC) \leftarrow -(ACC)$.
9. SSP : $(ACC) \leftarrow |(ACC)|$.
10. SSM : $(ACC) \leftarrow -|(ACC)|$.
11. TZE,o,B : Next P.I. is taken from location B if $(ACC) = \pm o$.
12. TPL,o,B : Next P.I. is taken from location B if $(ACC) \geq +o$.
13. TMI,o,B : Next P.I. is taken from location B if $(ACC) \leq -o$.
14. TNZ,o,B : Next P.I. is taken from location B if $(ACC) \neq \pm o$.

15.	ETRA,I,B	: Departure is made from the interpretive mode and control is sent to location B.
16.	ETZE,I,B	: Same as 15 if (ACC)= $\pm$ o.
17.	ETMI,I,B	: Same as 15 if (ACC)$\leq$ - o.
18.	CLS1,I,B	: (ACC1)$\Leftarrow$ -(B).
19.	CLA1,I,B	: (ACC1)$\Leftarrow$ (B).
20.	STO1,I,B	: (B)$\Leftarrow$ (ACC1).
21.	DVH2N,o,N	: (ACC)$\Leftarrow$ (ACC)/2^N N$\geq$ o.
22.	MPY2N,o,N	: (ACC)$\Leftarrow$ (ACC)$\cdot 2^N$, N$\geq$ o.
23.	STZ,I,A	: (A)$\Leftarrow$ o.
24.	ATOB,I,B	: (ACC)$\Leftarrow$ (ACC)**(B), $\lvert(B)\rvert < 2^{35}$.
25.	DPEXIT	: Departure is made from the interpretive mode and control passes to the next instruction in sequence.
26.	DPFLTI	: (ACC) is replaced by the M.P.N. equivalent of (ACC1)i.
27.	DPFLTF	: (ACC) is replaced by the M.P.N. equivalent of $(ACC1)_f$.
28.	DPTOM	: (ACC1) is replaced by the machine floating equivalent of (ACC). Numbers out of machine range are replaced by extreme values, i.e., o or $(377777777777)_8$.
29.	DPSQRT	: (ACC) is replaced by the square root of $\lvert$ (ACC) $\rvert$.
30.	DPEXP	: (ACC) is replaced by the exponential of (ACC). The argument must be less than 2^{b-1} ln2 in absolute value. For b=9 the absolute value of the argument must be less than 177.4.

31. DPLNX : (ACC) is replaced by the natural logarithm of (ACC). If the argument is less than or equal to zero, no computation is done, and the (ACC) is unchanged.

32. DPCOS : (ACC) is replaced by the cosine of (ACC). The argument plus $\pi/2$ must be less than 2^{35} in absolute value.

33. DPSIN : (ACC) is replaced by the sine of (ACC). The absolute value of the argument must be less than 2^{35}

34. DPATN : (ACC) is replaced by the arc tangent of (ACC). The result is between $-\pi/2$ and $\pi/2$.

35. DPASIN : (ACC) is replaced by the arc sine of (ACC). The absolute value of the argument must not be greater than 1.

36. DPACOS : (ACC) is replaced by the arc cosine of (ACC). The absolute value of the argument must not be greater than 1.

37. DPSINH : (ACC)$\leftarrow$ hyperbolic sine of (ACC).

38. DPCOSH : (ACC)$\leftarrow$ hyperbolic cosine of (ACC).

39. LDQ,I,A : (Pseudo-MQ)$\leftarrow$ (A).

40. STQ,I,A : (A)$\leftarrow$ (pseudo-MQ).

41. CACMQ : Next P.I. taken if (ACC)$>$(pseudo-MQ). Skip one P.I. if (ACC) = (pseudo-MQ). Skip two P.I.s if (ACC)$<$(pseudo-MQ).

42. DPFBCD,I,A : (ACC) is replaced by the M.P.N. equivalent of the BCD number stored starting at location A. This P.I. provides a way of entering constants into a code. An example is given in the section on input.

43. GET3,I,R : (ACC)$\leftarrow$ (R).

44. PUT3,I,R : (R)$\leftarrow$ (ACC).

45. GET1,I,R : (ACC1) $\leftarrow$ (R).

46. PUT1,I,R : (R) $\leftarrow$ (ACC1).

47. TXI,D,B : Index register 1 is incremented by D and the next P.I. is taken from location B.

Pseudo-instructions 43, 44, 45, and 46 above provide a crude way of communicating with MPP when the package is assembled as a relocatable subprogram. R is the address, relative to zero, of some 1 or 3 cell value within MPP.

Pseudo-instructions 48 through 55 listed below have proven of little value but are included for the benefit of codes already written. The function of each P.I. is described by giving the machine instruction which the P.I. simulates. K=0, 1, or 2. K may also be 3, 5, 6, or 7 on the 7094.

48. TIX,D,B : TIX B,1,D

49. TXH,D,B : TXH B,1,D

50. TNX,D,B : TNX B,2,D

51. TXL,D,B : TXL B,2,D

52. LXD,K,B : LXD B,K

53. SXD,K,B : SXD B,K

54. LXA,K,B : LXA B,K

55. SXA,K,B : SXA B,K

Example: Given the argument x in the ACC, compute

$$-16xe^{-x/4}\sqrt{\text{arc tan}(\ln x^2)}$$

and leave the result in the ACC. Assume $|x| \geq 1$.

```
YFUN SXD YFUN+4,4           SAVE INDEX 4
     TSX ENTRY,4            ENTER INTERPRETIVE
         STO,0,X            SAVE ARGUMENT
         MPY,0,X            SQUARE ARGUMENT
         DPLNX              LN OF X**2
         DPATN              ARC TANGENT
         DPSQRT             SQUARE ROOT
         STO,0,T            SAVE PART OF RESULT
         CLS,0,X
         DVH2N,0,2          -X/4
         DPEXP              EXPONENTIAL
         MPY,0,T
         MPY,0,X
         MPY2N,0,4
         CHS
         DPEXIT             LEAVE INTERPRETIVE
     LXD YFUN+4,4
FINIS TXI ENTRY,4,-1        GO GET NEXT P.I.
   X BSS 3
   T BSS 3
```

The above coding could be executed with the P.I.: YFUN.

Method and Accuracy For Elementary Function Pseudo-Instructions

The argument (ACC) $= x = F \cdot 2^p = g \cdot 10^q$, $.1 \leq |g| < 1$

1. Square Root:

An initial approximation y_o is computed as follows:

Case 1: p even, $y_o = \frac{1}{2}(\ |F|\ + 1\ ) \cdot 2^{p/2}$

Case 2: p odd, $y_o = \frac{1}{2}(\ |F|\ + \frac{1}{2}\ ) \cdot 2^{\frac{p+1}{2}}$

The square root is then y_5 obtained from the recursion formula

$$y_{n+1} = \tfrac{1}{2}(y_n + \frac{|x|}{y_n}).$$

The result should be correct to .3(105-b) significant decimal digits.

2. Exponential:

Put $x/\ln 2 = I + f$, I an integer and $o \leq |f| < 1$.

Then $e^x = 2^I \cdot e^{f \ln 2}$

$e^{f \ln 2}$ is computed from the exponential series. The result should be good to .3(105-b)-1 significant decimal digits if $|x| \leq 1$. If $|x| > 1$, the result should be correct to .3(105-b)-(q+1) significant decimal digits.

3. Natural Logarithm

$\ln x = \ln F + p \ln 2.$

$$\ln F = 2\left\{\left(\frac{F-1}{F+1}\right) + 1/3\left(\frac{F-1}{F+1}\right)^3 + 1/5\left(\frac{F-1}{F+1}\right)^5 + \ldots\right\}.$$

The result should be correct to .3(105-b)-1 significant decimal digits, except when a loss of significance occurs in computing F-1.

4. Sine and Cosine

Put $x/\pi/4 = I + f$, I an integer and $o \leq |f| < 1$.

$\sin x = \sin\left(I \cdot \frac{\pi}{4} + f \cdot \pi/4\right).$

Either $\sin f \cdot \pi/4$ or $\cos f \cdot \pi/4$ is computed from the sine or cosine series.

Cos x is obtained by computing $\sin(x + \pi/2)$.

The accuracy of sin x and cos x should be the same as that given for the exponential function, except for cos x when a loss of significance occurs in computing $x + \pi/2$.

5. Arc Tangent

The following relations are used.

$$\text{arc tan } x = \sum_{n=o}^{\infty} (-1)^n \frac{x^{2n+1}}{2n+1} = \frac{\pi}{2} - \text{arc tan}\left(\frac{1}{x}\right) = -\text{arc tan}(-x)$$

$$= \text{arc tan } a + \text{arc tan } \frac{x-a}{1+ax}.$$

The result should be correct to .3(105-b)-1 significant decimal digits.

6. Arc Cosine

arc cosine x = arc tangent ($\sqrt{1-x^2}$ /x), $1 \geq |x| > o$; arc cosine o = $\pi/2$.

The result should be correct to .3(105-b)-1 significant decimal digits, except when significance is initially lost in computing $1-x^2$.

7. Arc Sine

arc sine x = $\pi/2$ -arc cosine x.

8. Hyperbolic Cosine and Hyperbolic Sine

$\cosh x = (e^{x} + e^{-x}) /2.$

$\sinh x = (e^{x} - e^{-x}) /2.$

Results should have the same accuracy as that described for the exponential function, except for sinh x when x is close to zero and loss occurs in subtraction.

9. x^{β} with the P.I.: ATOB,I,B

If (B)=β has n significant binary bits and k of these are non-zero than x^{β} is computed using n+k-2 multiplies. One division is required if β is negative. This P.I. is used only for raising a multiple precision number to a integral power. $o^{\beta}=o$, $x^{o}=1$.

Note: It is the user's responsibility to insure that the argument x is in the proper range before computing $\sqrt{x}$, lnx, sin x, cos x, $\sin^{-1}x$, $\cos^{-1}x$.

Significance and Number Range Table

Let a multiple precision number have the following representation:

$$N = f \cdot 2^{p} = F \cdot 10^{q}, \quad \tfrac{1}{2} \leq |f| < 1, \text{ or } f = o,$$

$$.1 \leq |F| < 1, \text{ or } F = o.$$

The following table gives a few values of b, the approximate upper limits for $|p|$ and $|q|$ and number of significant decimal digits in the number. As described previously, b is the number of bits allocated for the binary power of two of a M.P.N.

b	$\|p\|$ must be less than	$\|q\|$ must be less than	Number of Significant Decimal Digits
9	256	78	28.8
10	512	155	28.5
11	1024	309	28.2
12	2048	617	27.9
13	4096	1234	27.6
14	8192	2467	27.3
15	16384	4933	27.0
16	32768	9865	26.7
17	65536	19729	26.4

A few octal representations of numbers when b = 9 are:

$1 = \frac{1}{2} \cdot 2 =$	200000000000	000000000000	000000000401
$-5=-(\frac{1}{2}+1/8)\cdot 2^{3}=$	640000000000	400000000000	400000000403
$.3=(.6)\cdot 2^{-1} =$	231463146314	314631463146	146314632377

Errors Detected

There are 6 types of serious errors detected in the package.

E1: An error condition exists if the unnormalized fractional part of a product is less than $\frac{1}{4}$ in absolute value. The error cannot occur if only normalized M.P.N.s are used.

E2: Overflow which may occur in several parts of the package or in the program of the user. Use of the MODIFY subroutine will usually correct overflow problems. Results which underflow are automatically set to zero.

E3: Division by zero is not allowed.

E4: Division by an unnormalized M.P.N. is not allowed.

E5: An error condition exists if an end of file is read by the input subroutine. The input tape will be repositioned in front of the end of file before taking the error exit.

E6: Some format errors are detected during output.

If one of the above errors is detected, a message is written on the output tape and the instruction STR is executed. The STR will normally cause control to go to the dump routine of IBSYS.

Error Exit Modification

Any or all of the above error exits may be modified by use of the sequence:

```
α     TSX    SETERS,4
α+1   PZE    EXIT1
α+2   PZE    EXIT2
α+3   PZE    EXIT3
α+4   PZE    EXIT4
α+5   PZE    EXIT5
α+6   PZE    EXIT6
α+7   PZE    NOBSR
α+8   TRA    CEXIT
α+9   Return
```

This sequence sets error Ei to transfer control to EXITi and the error message is not printed. Ei is unchanged if EXITi=0. If NOBSR≠0 then the tape is not backed over the end of file during E5. If the contents of $\alpha+8$ are not zero then the STR exit is replaced by the contents of $\alpha+8$.

Under the FORTRAN Monitor under IBSYS, it may be preferable to have $(\alpha+8)$= CALL EXIT in order to avoid returning to IBSYS and dumping.

Subroutines in the Package

1. Number Range Modification Subroutine - MODIFY

If it is necessary to have a greater range of numbers than is possible when b = 9, then b can be increased by giving the calling sequence

```
α      TSX MODIFY,4
α +1   PZE B
α +2   return
```

This subroutine sets up the entire package so that future computations will have (105-B) bits for the fractional part of numbers and B bits are used for the power of 2 exponent.

There may be constants outside this package which have to be modified also. This can be accomplished by inserting a transfer into MODIFY + 48 before giving the above calling sequence. This transfer should be to a code which will, for each M.P.N. to be modified, clear and add logical the third word of the M.P.N. and TSX ADJUST,4. Control will return to the location following the TSX, at which point the accumulator should be stored logically back into the third cell of the M.P.N.

2. Subroutine for Solution of Linear Equations - EQS

This subroutine solves a system of N simultaneous linear equations by the method of elimination with pivot selection.

To solve AS=B for S the following calling sequence is given.

```
α      TSX EQS,4
α +1   PZE A,O,S
α +2   PZE N              (N= number of equations)
α +3       error return (singular matrix)
α +4       normal return
```

A, B, and S are stored as follows:

a_{ij} is stored in location A+3(j-1+(i-1)(N+1)).
b_k is stored in location A+3(k(N+1)-1), k=1,2,...,N.
s_m will be stored in location S+3(m-1), m=1,2,...,N.

The matrix A is destroyed during the computation.

It should take about 2 minutes to solve 50 equations on the 7090.

3. <u>Input Subroutine - DPLOAD</u>

This subroutine reads and converts BCD information from SYSIN1. Records consisting of DEC,OCT,BCD,BCI, or TRA type information are allowed. In addition, if column 1 is not punched with D and columns 8, 9, and 10 are blank then DEC is understood.

M.P.N. records are also handled by this subroutine, and they are identified by having a D in column 1 followed by the multiple precision numbers starting in column 2. Multiple precision numbers are written exactly as numbers on DEC records, except up to 30 significant decimal digits may be written for each number; and the absolute value of the decimal power of ten exponent (n) can be any integer less than one third the core size minus the number of digits to the right of the decimal point in the fractional part of the number. A table of conversion constants for $|n| \leq 68$ is provided. If $|n| > 68$ the conversion constant is computed, requiring extra conversion time and may introduce round off errors.

All integers are converted exact by this subroutine, while non-integers are correct (unrounded) to the full significance in effect at time of conversion.

Loading is accomplished by giving the calling sequence

α	TSX DPLOAD,4
α +1	P β,0,γ
α +2	return

where P is either PZE or TXL.

OCT, BCD, DEC, BCI, or D type records are read and converted consecutively until the storage is filled from β through γ . It must be remembered that multiple precision numbers take 3 cells while all other types take one cell each. A TRA $\pm$L,I record will terminate loading and control will be transfered as specified by the TRA. Also a TOC $\pm$L,I is recognized while loading and has the same function as TRA except that L is treated as an octal integer. Records may be up to 120 characters in length.

DEC,OCT, BCI, and BCD records may have a decimal address, or the character equals (=) followed by an octal address, in columns 1 through 6 of the record. In either case the loading of data begins at the new address.

DPLOAD checks for a few clerical errors which may be in the data field of DEC and D type records. If P of the calling sequence is zero the machine stops after detecting an error. If P = 7, the machine does not stop on error detection.

Each error detected in a record causes the entire record to be printed on line with the column number in error and the number of records which have been read up to this point. If the machine stops (i.e., P = o) after an error print it may be restarted and loading will continue as if P = 7.

After reading a block of data with DPLOAD, a check may be made to determine if any errors were detected. This is accomplished by giving the instruction

α TSX TEST,4

Control returns to α +1 with the accumulator address containing the number of records read by DPLOAD since the last time TEST was entered. The accumulator sign is positive if no errors were detected, negative if one or more errors were detected.

The errors detected, and actions taken if the program is continued are listed below.

Error	Action Taken
1. Illegal character (i.e., anything other than o,1,...,9,E,.,+,-,dash,B, or comma)	ignored
2. 2 consecutive decimal points	2nd point ignored
3. 2 consecutive commas	zero separates the commas
4. point followed by plus	plus followed by point
5. point followed by minus	minus followed by point
6. point followed by dash (dash is treated as minus)	dash followed by point
7. comma followed by blank	comma followed by zero, followed by blank

(continued)

Error	Action Taken
8. an operation in columns 8, 9, 10 other than DEC,OCT,BCD,TRA,BCI, or TOC	entire record ignored

Example

Suppose b=9, and the following calling sequence was given starting at $(50060)_8 = (20528)_{10}$.

```
TSX DPLOAD,4
TXL INPUT,0,-1
```

The 14 cards listed below are to be read from SYSIN1.

```
       BCI   1,+40000
       BCD 1000000
           257X
       DEC ..-1,,240
       BDC 1IDENT
D-.3141592653589793238462643 39C28E01
DO.
       DEC 2.,0,
   448 OCT 460000000355
  =611 OCT 450100000437,060100000623
       TOC 50060
       TRA 20528
       TOC 0,4
       TRA 2,4
```

An on-line print out, except for extra spacing, would appear as follows.

```
       DEC 257X                            CHECK CARD 3 IN COLUMN 15
       DEC ..-1,,240                       CHECK CARD 4 IN COLUMN 13
       DEC ..-1,,240                       CHECK CARD 4 IN COLUMN 14
       DEC ..-1,,240                       CHECK CARD 4 IN COLUMN 17
       BDC 1IDENT                          CHECK CARD 5 IN COLUMN  8
D-.31415926535897932384626433 9C28E01      CHECK CARD 5 IN COLUMN 31
       DEC 2.,0,                           CHECK CARD 7 IN COLUMN 17
```

core storage, starting at symbolic location INPUT, would contain the following octal words.

```
200400000000 000000000000 000000000401 575631463146 000000000000
000000000360 711037552421 413214106461 461461510402 000000000000
000000000000 000000000000 202400000000 000000000000 000000000000
```

The 3 octal words on the ninth and tenth cards would be stored in locations $(488)_{10}$, $(611)_8$ and $(612)_8$, and control would go to octal location 50062.

Note that when an entire record is rejected the record(card) count is not incremented.

The variable field for BCI,OCT,DEC,TRA, and TOC type records may begin anywhere in columns 12 though 16.

Input by DPLOAD is accomplished by use of the subroutine RTX (described below) which in turn uses IOEX. All input is single record buffered so it is recommended that the user's data deck have a dummy data card at the end. However, an end of file will usually be sufficient.

DPFBCD

DPLOAD is used for execution of the pseudo instruction DPFBCD,I,A. The conversion is accomplished by forcing DPLOAD to get its input from storage rather than SYSIN1.

Only one number can be converted by DPFBCD by a single use. The BCD number is written as D type data and the number must be preceded by the character D and terminated by a blank.

Constants can be entered into the code in BCD form and converted in this manner as an initial operation following any use of the MODIFY subroutine.

Example for DPFBCD

The following code could be used to enter the constants 1.,$-\pi/4$, and ln2 into storage locations ONE, MPIOV4, and LOG2.

```
        TSX     ENTRY,4
                DPFBCD,0,C2
                STO,0,LOG2
                DPFBCD,0,C1
                STO,0,ONE
                DPFBCD,0,C3
                STO,0,MPIOV4
                DPEXIT
         .
         .
         .
C1      BCI     2,D10E-1
C2      BCI     6,D.0693147180559945309417232lE1
C3      BCI     6,D-.785398163397448309615660845 8
LOG2    BSS     3
MPIOV4  BSS     3
ONE     BSS     3
```

<u>Reading Errors</u>

Noise records (records of length less than 3 words) are ignored. IOEX prints a message if a record is permanently redundant and the record as read the last time is accepted. Reading through an end of file is considered an error, but this can be changed by use of the SETERS routine.

4. <u>Input Record Scan Deletion Subroutine - NOSCAN</u>

The DPLOAD routine can be set to omit error prints 1. through 8. by giving a TSX NOSCAN,4 but the actions taken on these errors are the same as described before.

5. BCD Read Subroutine - RTX

This subroutine reads BCD input records from SYSIN1. The calling sequence for RTX is as follows.

```
α      TSX    RTX,4
α+1    PZE    A,O,EOFR
α+2    Return
```

One BCD record or the first 22 words of a BCD record is read into core storage starting at symbolic location A.

Records are single record buffered automatically. An end of file causes control to be sent to symbolic location EOFR. If EOFR were *-1 or 0 then end of files would be ignored with buffering initialized at the beginning of each file to be read.

RTX uses IOEX for reading. Noise records are ignored. Permanent redundancy causes IBSYS to print a message and then RTX will accept the record as read the last time.

Warning

Special care is required when non-IOEX input/output routines are used in conjunction with MPP input/output. The FORTRAN II library routines are examples of a type which will cause conflict. Such routines may be used only after the channels have been freed by IOEX. The sequence

```
TCOX *
TCOX *-1
```

for each channel X is sufficient.

6. Binary to Decimal Conversion and Output

a. Output Subroutine - DPBDC3

This subroutine converts information in core and prints it on SYSOU1. Binary integers may be printed in octal or decimal form. Multiple precision numbers may be printed in floating or fixed point decimal form. Integers are exact while multiple precision numbers are rounded to a specified number of places. BCD information can be printed also.

Output is accomplished by giving the calling sequence

```
α       TSX     L,4
α+1     PZE     A,0,B
α+2     Return
          .
          .
          .
L       TRA     DPBDC3
        BCI     n,format
```

The block of numbers in continuous core storage from A through B will be printed as specified by the format and control returns to α +2. L is any core location and the BCD format (see format description) may be of any length.

Example: The following code would cause 100 full word integers stored starting at location Y to be printed as decimal integers.

```
Q       TSX     F,4
Q+1     PZE     Y,o,Y+99
Q+2     Return
          .
          .
          .
F       TRA     DPBDC3
        BCI     1,8I14
```

b. <u>Output Subroutine - D(BDC)</u>

This subroutine handles conversion in exactly the same way as DPBDC3. The only difference is in the way information is picked up from core for conversion and printing. D(BDC) allows printing of information which is not stored sequentially in core.

Output is accomplished by giving the calling sequence

```
α       TSX    L,4
α+1     PRF    OP,I,A
         .
         .
         .
α+k     PRF    OP,I,A
        TRA    D(FIL)
         .
         .
         .
L       TRA    D(BDC)
        BCI    n,format
```

The user's coding in α+1 through α+k determines which numbers are to be converted and printed as specified by the format. See format description.

The block of coding in α+1 through α+k may not contain any instructions other than the following types.

i. Any machine instruction which does not alter sequential flow of control. For example, test and unconditional transfer instructions are not allowed.

ii. Any of the instructions TXI, TIX, TXH, TNX, TXL.

iii. The STR instruction, which has the pseudo operation CLA,I,A in its address, tag, and decrement portions. This is the instruction used to pick up numbers for conversion and printing.

Index register 4 may not be used in the above instructions. All other index registers are available, but only 1 and 2 are saved and restored.

Example

The following code would cause the single celled number N and the 3 celled numbers Bi, i = 1,2,...,(N) to be printed according to some format.

```
              TSX    FORM,4
              STR    CLA,O,N
              LXA    N,1
              AXT    0,2
              STR    CLA,2,B
              TXI    *+1,2,-3
              TIX    *-2,1,1
              TRA    D(FIL)
               .
               .
               .
        FORM  TRA    D(BDC)
              BCI    n,format
               .
               .
               .
           N  BSS    1
           B  BSS    30
```

Note

The store and trap (STR) instruction used in the calling sequence is not executed, but is used to isolate types i and ii instructions from the data pickup instruction. All instructions between the TSX and TRA D(FIL) are handled interpretively.

Warning

Special care is required when non-IOEX input/output routines are used in conjunction with MPP input/output. The FORTRAN II library routines are examples of a type which will cause conflict. Such routines may be used only after the channels have been freed by IOEX. The sequence

```
TCOX *
TCOX *-1
```

for each channel X is sufficient.

Format Description

A format consists of any number of field specifications. The general field specification is of the following form.

$\alpha S \beta (\gamma) gPnTd \cdot wZ$

α, β, γ, g, n, d, and w are positive integers.

T is the type conversion character and T is I, E, F, O, A, or X.

Z is a slash, comma, or blank, and Z serves as a field specification separator.

Each non-numeric Hollerith character in a field specification indicates what is to be done with the integer just preceding it.

Each field specification is executed n times and n data fields are printed, each having a total column width w with the spread of information being determined by α, β, γ, g, and d.

T = I specifies full word binary to decimal integer conversion.

T = E specifies binary to floating point decimal conversion of multiple precision numbers.

T = F specifies binary to fixed point decimal conversion of multiple precision numbers.

T = O specifies full word binary to octal conversion.

T = A specifies BCD to BCD conversion.

T = X specifies insertion of blank characters.

If either of the integers α, β, γ, or g is zero then that integer and the following Hollerith character may be omitted.

The format is scanned from left to right. Conversion as indicated by a specification is completed before checking on the separator Z. If Z is a blank, then all information converted at this point is printed and if there is still more information to be converted and printed, scanning goes back to the beginning of the format. If Z is a slash, all information converted at this point is printed and scanning for the next line starts immediately after the slash. K consecutive slashes separating two specifications cause K-1 blank lines to be printed. If Z is a comma, conversion continues with the specification following the comma being scanned.

Binary to Decimal Integer Conversion (T=I) $\{gPnIwZ\}$

n single celled binary integers are converted and printed as decimal integers, each with a total column width w; and each number is enclosed in parenthesis if $g \neq o$. The restrictions are $n > o$, $w \leq 26$.

Example: Suppose the binary equivalent of the integers 1, -2, 3, 7000, and -56789 are in consecutive storage starting at location C. Then the sequence:

```
         TSX FORM,4
             C,o,C+4



    FORM TRA DPBDC3
         BCI 2,1P1I4,4I5
```

would cause the following line to be printed.

```
 (1)   -2    3 700056789
```

Note that the negative sign of the last number would not be printed since w = 5.

Binary to Floating Point Decimal Conversion (T=E) $\{\alpha S \beta (\gamma) gPnEd \cdot wZ\}$

n 3 cell multiple precision numbers are converted and printed, each number having a total column width w. Information within the column is arranged as follows from left to right.

1. $w - \alpha - \beta - \gamma - d - 2$ blanks.

2. q, the decimal exponent of the number occupying β places and in parenthesis if $g \neq o$.

3. The sign of the number (blank for +, or a minus sign).

4. α significant digits of the number.

5. A decimal point.

6. d significant decimal digits (rounded).

7. q, the decimal exponent of the number occupying γ places and in parenthesis if $g \neq o$.

Restrictions are $n > o$, $w \geq \alpha + \beta + \gamma + d + 2$, $\beta \leq 26$, $\gamma \leq 26$, $d + \alpha < 30$.

Example: The format 1S4(1P1E5.13,3)1E10.19 would cause the binary equivalent of 12345.67 and -.00765432 to be printed as follows.

(4) 1.23457 -.7654320000 -2

Binary to Fixed Point Decimal Conversion (T=F) $\{\alpha S nFd \cdot wZ\}$

n 3 cell multiple precision numbers are converted and printed as fixed point decimal numbers, each number having a total column width w. Information within the column is arranged from left to right as follows. Before conversion, the number is multiplied by 10^{α}

1. The integral part of the number (with sign) occupying w-d-1 spaces.

2. A decimal point.

3. d decimal places (rounded) of the fractional part of the number.

Restrictions are $n > o$, $w \geq d + 2 +$ number of digits in integral part.

This subroutine arbitrarily refuses to print fixed point multiple precision numbers whose absolute value is greater than $2^{35}-1$. If such a number is found, the specification is treated as E type rather than F type and the number is printed in floating point form, but the power of ten exponent cannot be printed unless either β(or γ) is written in the specification. β and γ are ignored in F type conversion, so no harm is done by specifying one of the two just in case large numbers are printed.

Example: The format 3)3F10.20 would cause the binary equivalent of -101.01, 10^{40}, and .00004778966 to be printed as follows.

-101.0100000000 .1000000000 41 .0000477897

Binary to Octal Conversion (T=O) $\{\alpha S \beta (\gamma) gPnOd.wZ\}$

n* single celled binary integers are converted and printed as octal integers, with locations if desired. Each number has a total column width w and information within the column is arranged from left to right as follows.

1. $w-\alpha-\beta-\gamma-$ d blanks.

2. The location as a decimal integer occupying β places and in parenthesis if g = 1, 3, 9, or 11.

3. The location as an octal integer occupying γ places and in parenthesis if g = 2, 3, 10, or 11.

4. α blanks.

5. The last d octal digits of the number.

Restrictions are $d \leq 12, \beta \leq 26, \gamma \leq 26, w \geq \alpha + \beta + \gamma + d, n > o$. If $g \geq 8$, then as far as the calling sequence is concerned, the number described above is not counted as having been printed; thus allowing a single cell to be printed both in octal and decimal with or without decimal and/or octal locations; or if d = 0 allowing a location to be printed with either a single celled number or multiple precision number.

Example: The format 1S5(9P1Oo.7,1I6,2S3)1O12.21,1O3.7 would cause the octal numbers -17,314631463146 and 7777777 in decimal locations 100, 101, and 102 to be printed as follows.

(100) -15 145 314631463146 777

* If n > 1 then g must be less than 8.

BCD Conversion (T=A) $\{nAwZ\}$

n BCD words are printed as n BCD fields with each field having a total column width w. If w > 6, then each field consists of w-6 blanks followed by the BCD word consisting of 6 characters. If $w \leq 6$ then the field consists of the left most w characters of the BCD word.

Blank Insertion (T=X) $\{nXZ\}$

n blanks are inserted into the print line.

BCD Characters Within the Format

All of the previously described field specifications may be preceded by iC followed by i Hollerith characters. The i characters will be printed just before the first number printed by the specification which follows.

Example: Given the binary equivalent of the integer 2, and multiple precision numbers .006, -10^{15}, -10^{-8}, and 10^{6} starting in decimal core location 100; the calling sequence

```
            TSX FPR,4

                100,0,112

            .
            .
            .

        FPR TRA DPBDC3

            BCD 623C  TEMPORIES STARTING AT5(8P10o.6/

            BCD 51I4,3)2F4.1o,1P1S5(2E5.16
```

would cause the following two lines to be printed.

```
        TEMPORIES STARTING AT  100

          2     .0060 -.1000 16    (-8)-1.00000     (6) 1.00000
```

BCD Tape Write Subroutine - WOT

This subroutine is used by MPP for all of its printing of BCD records on SYSOU1. Output is single record buffered and IOEX is trusted to write the record correctly. If the end of tape is sensed a message is printed and the machine pauses for a fresh SYSOU1.

<u>Printing of Tables</u> $\{mK\alpha\ S\beta(\gamma)gPnTd\cdot wZ\}$

To make numbers with many digits more readable, an additional feature is provided in the format specification for E and F type conversion. Either the F or the E specification as described previously may be preceded by mK.

The effect of mK is to cause blocks of m digits to the right of the decimal point to be separated by one blank. The separation will occur on both E and F type, and it will continue until reset by another mK appearing in some specification.

When this feature is used, the column width w must be increased enough to allow j extra spaces.

Let $\frac{d}{m} = k + R$, k an integer and $o \leq R < m$.

$$\text{Then } j = \begin{cases} o \text{ if } m=o. \\ k \text{ if } R>o. \\ k-1 \text{ if } R=o. \end{cases}$$

<u>Example</u>: Given Xo,DX,N,K

1. Compute πe^{Xi} and arc tan (Xi), Xi = Xo+(i-1)DX
   i = 1,2,...,N

2. Print a table of Xi , πe^{Xi} , arc tan (Xi)

3. If $K \neq \pm o$ then use MODIFY to reset the package so that $|K|$ bits are used for the power of two exponent.

4. If $K \leq -o$ give an octal dump of arc tan (Xi) , i = 1,2,...,N.

5. If Xo = DX = o then terminate the job by CALL EXIT.

6. MPP to be used is a relocatable subprogram which operates under FORTRAN II under IBSYS.

```
$EXECUTE       FORTRAN
$ID   THECODER*001/63/003$EXAMPLE OF MPP USE
*      XEQ
*      FAP
*EXAMPLE OF USE OF MPP
       COUNT    90
       EXTERN   ENTRY,CLA,STO,DPFBCD,DPEXP,MPY,DPATN
       EXTERN   DPLOAD,MODIFY,TEST,EXIT,ADD,ETRA,DPEXIT
       EXTERN   D(BDC),D(FIL),DPBDC3
 START TSX      DPLOAD,4               READ N AND K
       TXL      N,O,K                  DONT STOP CN ERROR
       CLA      K
       TZE      NOMOD
 MOD   STA      *+2
       TSX      MODIFY,4               MODIFY MPP TO
       PZE      **                     USE N BITS.
 NOMOD TSX      DPLOAD,4               READ XO AND DX
       TXL      XO,O,DX+2
       TSX      TEST,4                 CHECK CARD ERRORS
       TMI      EXIT                   YES---EXIT
       CLA      DX                     NO
       TNZ      AA
       CLA      XO
       TNZ      AA
       CALL     EXIT                   DX=AO=O--SO EXIT
 AA    LXA      N,1
       AXT      O,2
       TSX      ENTRY,4                ENTER INTERPRETIVE
                DPFBCD,O,PII           REPLACE BCD PI
                STO,O,PI               BY M.P.N. PI
                DPEXIT                 LEAVE INTERPRETIVE
 BB    TSX      ENTRY,4
                CLA,O,XO
                STO,2,ARGT             ENTER X(I) IN TABLE
                DPEXP                  EXP(X(I))
                MPY,O,PI
                STO,2,EXPT             ENTER EXP IN TABLE
                CLA,O,XO
                DPATN                  ATAN(X(I))
                STO,2,ATNT             ENTER ATAN IN TABLE
                CLA,O,XO
                ADD,O,DX               INCREMENT
                STO,O,XO
                ETRA,O,*+1             LEAVE INTERPRETIVE
       TXI      *+1,2,-3
       TIX      BB,1,1
*               PRINT THE TABLE
       TSX      HEAD,4                 START HEAD PRINT
       TRA      D(FIL)                 END PRINT
```

```
       TSX    TFORM,4               START TABLE PRINT
       LXA    N,1
       AXT    O,2
 LOOP  STR    CLA,2,ARGT            PICK UP X
       STR    CLA,2,EXPT            PICK UP PI*EXP(X)
       STR    CLA,2,ATNT            PICK UP ATAN(X)
       TXI    *+1,2,-3
       TIX    LOOP,1,1              GO BACK
       TRA    D(FIL)                END TABLE PRINT
*             CHECK FOR OCTAL DUMP
       CLA    K
       TPL    START                 NO DUMP IF PLUS.
       PXD    O,2
       PDC    O,2
       TXI    *+1,2,ATNT-1
       SXD    *+4,2
       TSX    DHEAD,4               HEAD THE DUMP
              10,0,10
       TSX    DFORM,4               PRINT THE DUMP
              ATNT,O,**             **=ATNT+N-1
       TRA    START                 GO BACK FOR NEW DATA
 HEAD  TRA    D(BDC)                PRINTS HEADING ON NEW PAGE
       BCI    8,1C1,12X1CX,15X,9CPI*EXP(X),20X,10CARC TAN(X)//
 TFORM TRA    D(BDC)
       BCI    6,1C 3)1F2.15,5K1S1P5(1E17.31,F17.27
 DHEAD TRA    DPBDC3
       BCI    9,1CO11X3CLOC5X,4C(...12X10CARC TAN(X),12X,4C...)
 DFORM TRA    DPBDC3
       BCI    5,3S8)2P1O12.32,2O12.15
 PII   BCI    6,D3.14159265358979323846264338328
 N     BSS    1
 K     BSS    1
 XO    BSS    3
 DX    BSS    3
 PI    BSS    3
 ARGT  BSS    3*2000
 EXPT  BSS    3*2000
 ATNT  BSS    3*2000
       END
*             (MPP PACKAGE GOES HERE)
*     DATA
       DEC    5,0                   N AND K
D-10.,5.                            XO AND DX
       DEC    8,-11                 N AND K
D-.90E2,40.                         XO AND DX
       DEC    0,0                   END JOB
DO.EO,O.                            END JOB
```

The above program and data would cause two pages of output as follows.

First page

X	PI*EXP(X)		ARC TAN(X)
-10.00	(-4)	1.42628 08581 53150 16	-1.47112 76743 03734 59
-5.00	(-2)	2.11678 84792 60429 67	-1.37340 07669 45015 86
0.00	(0)	3.14159 26535 89793 24	0.00000 00000 00000 00
5.00	(2)	4.66253 69033 27078 08	1.37340 07669 45015 86
10.00	(4)	6.91981 83125 51164 68	1.47112 76743 03734 59

Second page

X	PI*EXP(X)		ARC TAN(X)
-90.00	(-39)	2.57422 49862 95062 81	-1.55968 56728 97289 15
-50.00	(-22)	6.05934 63529 75874 74	-1.55079 89928 21746 09
-10.00	(-4)	1.42628 08581 53150 16	-1.47112 76743 03734 59
30.00	(13)	3.35725 50038 09131 03	1.53747 53309 16649 42
70.00	(30)	7.90248 36491 15328 54	1.55651 15842 07499 99
110.00	(48)	1.86012 82224 22199 02	1.56170 56681 29836 80
150.00	(65)	4.37846 77798 59209 79	1.56412 97588 91028 39
190.00	(83)	1.03062 68066 99454 58	1.56553 32174 97301 24

LOC	(...	ARC TAN(X)	...)
(51474)	707507436664	610642266434	433612752001
(51477)	706401123263	533216054600	722566142001
(51502)	674233645405	600051602725	576063242001
(51505)	304627767342	306340135451	171727346001
(51510)	307167426070	020602533113	015270442001
(51513)	307713742512	141612203176	027716456001
(51516)	310152635504	214501465665	261677306001
(51521)	310306621707	317331605432	265347206001

Reference List of Entries In MPP

Non-control Pseudo-operations

ADD	CACMQ	CHS	CLA	CLA1	CLS
CLS1	DVH	DVH2N	GET1	GET3	LDQ
MPY	MPY2N	PUT1	PUT3	SSM	SSP
STO	STO1	STQ	STZ	SUB	

Control Pseudo-operations

DPEXIT	ETMI	ETRA	ETZE	TMI	TNZ
TPL	TXI	TZE			

Elementary Function Pseudo-operations

ATOB	DPACOS	DPASIN	DPATN	DPCOS	DPCOSH
DPEXP	DPLNX	DPSIN	DPSINH	DPSQRT	

Data Transformation Pseudo-operations

DPFBCD	DPFLTF	DPFLTI	DPTOM

Miscellaneous Pseudo-operations

LXA	LXD	SXA	SXD	TIX	TNX
TXH	TXL				

Control Subroutines

ADJUST	ENTRY	MODIFY	NOSCAN	SETERS	TEST

Input Subroutines

DPLOAD	RTX

Output Subroutines

DPBDC3	D(BDC)	D(FIL)	WOT

Mathematical Subroutines

EQS

NBS TECHNICAL PUBLICATIONS

PERIODICALS

JOURNAL OF RESEARCH reports National Bureau of Standards research and development in physics, mathematics, chemistry, and engineering. Comprehensive scientific papers give complete details of the work, including laboratory data, experimental procedures, and theoretical and mathematical analyses. Illustrated with photographs, drawings, and charts.

Published in three sections, available separately:

- **Physics and Chemistry**

Papers of interest primarily to scientists working in these fields. This section covers a broad range of physical and chemical research, with major emphasis on standards of physical measurement, fundamental constants, and properties of matter. Issued six times a year. Annual subscription: Domestic, $5.00; foreign, $6.00*.

- **Mathematical Sciences**

Studies and compilations designed mainly for the mathematician and theoretical physicist. Topics in mathematical statistics, theory of experiment design, numerical analysis, theoretical physics and chemistry, logical design and programming of computers and computer systems. Short numerical tables. Issued quarterly. Annual subscription: Domestic, $2.25; foreign, $2.75*.

- **Engineering and Instrumentation**

Reporting results of interest chiefly to the engineer and the applied scientist. This section includes many of the new developments in instrumentation resulting from the Bureau's work in physical measurement, data processing, and development of test methods. It will also cover some of the work in acoustics, applied mechanics, building research, and cryogenic engineering. Issued quarterly. Annual subscription: Domestic, $2.75; foreign, $3.50*.

TECHNICAL NEWS BULLETIN

The best single source of information concerning the Bureau's research, developmental, cooperative and publication activities, this monthly publication is designed for the industry-oriented individual whose daily work involves intimate contact with science and technology—*for engineers, chemists, physicists, research managers, product-development managers, and company executives.* Annual subscription: Domestic, $1.50; foreign, $2.25*.

*Difference in price is due to extra cost of foreign mailing.

NONPERIODICALS

Applied Mathematics Series. Mathematical tables, manuals, and studies.

Building Science Series. Research results, test methods, and performance criteria of building materials, components, systems, and structures.

Handbooks. Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications. Proceedings of NBS conferences, bibliographies, annual reports, wall charts, pamphlets, etc.

Monographs. Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

National Standard Reference Data Series. NSRDS provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated.

Product Standards. Provide requirements for sizes, types, quality and methods for testing various industrial products. These standards are developed cooperatively with interested Government and industry groups and provide the basis for common understanding of product characteristics for both buyers and sellers. Their use is voluntary.

Technical Notes. This series consists of communications and reports (covering both other agency and NBS-sponsored work) of limited or transitory interest.

CLEARINGHOUSE

The Clearinghouse for Federal Scientific and Technical Information, operated by NBS, supplies unclassified information related to Government-generated science and technology in defense, space, atomic energy, and other national programs. For further information on Clearinghouse services, write:

Clearinghouse
U.S. Department of Commerce
Springfield, Virginia 22151

Order NBS publications from:
Superintendent of Documents
Government Printing Office
Washington, D.C. 20402

U.S. DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20230

OFFICIAL BUSINESS

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE