COMPUTATION CENTER
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139

TO:	CTSS USERS
FROM:	PROGRAMMING STAFF
SUBJECT:	CTSS LIBRARY SUBPROGRAMS NOT INCLUDED IN THE CTSS PROGRAMMER'S GUIDE
DATE:	JULY, 1967

Introduction

The following is an index by entry names of subprograms included in this memo. A user should refer to CC-174 for the usage of XSIMEQ and XDETRM. After the index are write-ups about the subprograms.

It is suggested that this annotated itemization be inserted in the CTSS Programmer's Guide after section AG.13.00.

	ENTRY NAME	SYNOPSIS
1	ACØS, ASIN	Arccosine, arcsine functions. Floating-point argument.
2	ATAN, ATN	Arctangent function. Floating point.
3	DETCS	Simulates a FORTRAN function call to XDETRM.
4	DFAD, DFSB, DFMP, DFDP, SFDP, DCEXIT	Performs double precision floating-point operations on numbers stored in consecutive storage locations.
5	DIM	Positive difference function.
6	EXP	Exponential function. Floating point (1).
7	EXP(1	Computes I**J. Fixed point (1).
8	EXP(2	Computes X**K. Floating-point number to a fixed-point power (1).
9	EXP(3	Computes X**Y. Floating arguments (1).
10	FINT MINT	Converts MAD integer to FORTRAN integer. Converts FORTRAN integer to MAD integer.
11	FLIP	Interchanges rows and columns in a MAD two-dimensional array.
12	INDV, DPNV	Integration of differential equations.
13	INT	Truncation. Floating point.
14	IØSET, IØPAR, IØEND, IØSCP, IØITR	Compiled into MAD routines during translations of FORTRAN statements involving iterated input-output.
15	LØG	Logarithm (base e) function.
16	MAX0	Maximum, fixed-point argument, floating-point function.
17	MAX1, XMAX0	Maximum, floating-floating, fixed-fixed.
18	MIN0	Minimum, fixed-point argument, floating-point function.
19	MIN1, XMIN0	Minimum, floating-floating, fixed-fixed.

	ENTRY NAME	SYNOPSIS
20	MØD	Remaindering. Floating point.
21	RANNØ, SETU	Random number generator.
22	SIMCS	Simulates a FORTRAN call to XSIMEQ.
23	SIN, CØS	Sine and cosine functions. Floating point.
24	(SLI)	Short list input. Used by FORTRAN routines.
25	(SLØ)	Short list output. Used by FORTRAN routines.
26	SQRT, SQR	Square root function. Floating point.
27	TAN, CØT	Tangent and cotangent functions. Floating point.
28	TANH	Hyperbolic tangent function.
29	XDIM	Finds the absolute difference between the two arguments. Fixed point.
30	XDTRM	Called by MAD and MADTRN programs instead of XDETRM.
31	XINT, XFIX	Truncation. Changes floating-point numbers to fixed-point numbers.
32	XLØC	Finds the location where a variable is stored.
33	XMAX1	Finds the maximum values of a set of floating-point numbers. Result is fixed-point.
34	XMIN1	Minimum floating arguments, fixed result.
35	XMØD	Remaindering. Fixed point.
36	XSIGN, SIGN	Transfers the sign of the second argument to the first.

	ENTRY NAME	SYNOPSIS
37	XSMEQ	Called by MAD and MADTRN programs instead of XSIMEQ.
38	.01300	Computes Y**Z. Floating point. Used by MAD routines.
39	.01301	Computes X**K. X floating point, K fixed point. Used by MAD routines.
40	.01311	Computes I**J. Fixed point. Used by MAD routines.
41	.03310, .03311	Computes linear subscripts for two-dimensional MAD arrays.

(1) See corresponding entries that are used by MAD routines, nos. 38-41.

CTSS SUBPROGRAM

Entry Name:	ASIN, ACØS
Purpose:	Computes the principal value of arcsine X or arccosine X for a single precision floating-point argument.
Length:	250 octal locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	The error condition is met and ERRØR subprogram called, if the absolute value of the argument is greater than 1.0. The ERRØR subprogram may also be called in case of machine failure. Upon return from ERRØR, the ASIN and ACØS functions send control to the LDUMP subprogram.

Calling Sequence:

FORTRAN	MAD	FAP
Y = ASINF(X)	Y = ASIN.(X)	CLA X TSX $ASIN, 4 STØ Y

where:

Y	is an angle in floating-point radians in the first and second quadrants for ACØS and the first and fourth quadrants for ASIN.
X	is a floating-point number between -1 and +1.

Reference: SHARE Distribution No. 670, IB ANS2 and IB ACS2

CTSS SUBPROGRAM

Entry Name:	ATAN, ATN
Purpose:	Computes the principal value of arc tangent X for any single precision floating-point argument with either entry.
Length:	127 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

```
     FORTRAN                 MAD                    EAP

     ANGLE = ATANF(TANG)     ANGLE = ATAN.(TANG)    CLA    TANG
                                                    TSX    $ATAN,4
                                                    STØ    ANGLE
```

where:

ANGLE	is an angle in floating-point radians in the first or fourth quadrant.
TANG	is the tangent of an angle in floating-point.

Reference: SHARE Distribution No. 507, IB ATN 1

CTSS SUBPROGRAM

Entry Name:	DETCS
Purpose:	Simulates a FORTRAN function call to XDETRM.
	This routine is used by the library program XDTRM.
Length:	26 octal locations
Transfer Vector:	XDETRM

Calling Sequence:

```
     MADTRN:   CALL DETCS (NFØR,NØFØR,A,D,MFØR)

     MAD:      EXECUTE DETCS.(NFØR,NØFØR,A,D,MFØR)
```

where:

NFØR	is a FORTRAN (decrement) integer giving the maximum number of rows or columns which the matrix A may have.
NØFØR	is a FORTRAN (decrement) integer giving the number of rows or columns in the matrix A at the time XDTRM is called.
A	is an unsubscripted floating-point variable referring to the matrix.
D	is a floating-point variable by whose value the value of the determinant will be modified. Upon return, D=Det(A).
MFØR	upon return will contain a FORTRAN (decrement) integer which will be

1	if the solution was successful
2	if an overflow occurred
or 3	if the matrix A is singular.

Execution:

Given the above calling sequence, DETCS reforms it in such a way as to simulate a FORTRAN function call. XDETRM is called by this routine. Upon return from XDETRM, the resulting integer value is stored in MFØR.

A programmer may call this routine instead of XDTRM if he will alter his integers in the call so they are FORTRAN integers (by multiplying each of them by 2.P.18, shifting the integers left 18 places, or calling FINT) and upon return changing the resulting FORTRAN integer MFØR to a MAD integer (by dividing it by 2.P.18, shifting the integer right 18 places, or calling MINT).

Restrictions:

All integers are FORTRAN (decrement) integers.

The matrix A must be a square matrix with the base element set at 1, that is, A(1,1)=A(1). However, in the call, A must not be subscripted.

Reference:

CC-174

CTSS SUBPROGRAM

Entry Names:	DFAD, DFSB, DFMP, DCEXIT, DFDP, SFDP
Purpose:	To perform double-precision floating-point operations on numbers stored in consecutive storage locations.
Length:	153 octal locations
Transfer Vector:	ENDJØB
Error Procedure:	If division by zero is attempted and DCEXIT has not been called, ENDJØB will be called. Floating-point operations may result in a floating-point trap.

Calling Sequences:

```
        FORTRAN:   CALL DFAD(ADDEND,AUGEND,SUM)
                   CALL DFSB(SUBTRA,MINUND,DIFF)
                   CALL DFMP(MLTPND,MLTIER,PRODCT)
                   CALL DFDP(DVDND,DVSOR,QTNT)
                   CALL SFDP(DVDND,SPDVSR,QTNT)
                   CALL DCEXIT(ERRLOC)

         MAD:      EXECUTE DFAD.(ADDEND,AUGEND,SUM)
                   EXECUTE DFSB.(SUBTRA,MINUND,DIFF)
                   EXECUTE DFMP.(MLTPND,MLTIER,PRODCT)
                   EXECUTE DFDP.(DVDND,DVSOR,QTNT)
                   EXECUTE SFDP.(DVDND,SPDVSR,QTNT)
                   EXECUTE DCEXIT.(ERRLOC)

         FAP:      TSX    $DFAD,4
                   OPN    ADDEND,TAG1
                   OPN    AUGEND,TAG2
                   OPN    SUM,TAG3

                   TSX    $DFSB,4
                   OPN    SUBTRA,TAG4
                   OPN    MINUND,TAG5
                   OPN    DIFF,TAG6

                   TSX    $DFMP,4
                   OPN    MLTPND,TAG7
                   OPN    MLTIER,TAG8
                   OPN    PRODCT,TAG9
        TSX    $DFDP,4
        OPN    DVDND,TAG10
        OPN    DVSOR,TAG11
        OPN    QTNT,TAG12

        TSX    $SFDP,4
        OPN    DVDND,TAG13
        OPN    SPDVSR,TAG14
        OPN    QTNT,TAG15

        TSX    $DCEXIT,4
        OPN    ERRLOC
```

where:

OPN	is any operation code which allows an address (and tag, if the tag position is used).
TAGI	is any index register to modify the operand, except index register 4.
ADDEND	is the location of the high-order part of the addend. The low-order part is at ADDEND+1(FAP), or if the high-order part is at ADDEND(2), then the low-order part is at ADDEND(1).
AUGEND	is the location of the high-order part of the augend.
SUM	is the location of the high-order part of the sum of the addend and the augend.
SUBTRA	is the location of the high-order part of the subtrahend.
MINUND	is the location of the high-order part of the minuend.
DIFF	is the location of the high-order part of the subtrahend minus the minuend.
MLTPND	is the location of the high-order part of the multiplicand.
MLTIER	is the location of the high-order part of the multiplier.
PRODCT	is the location of the high-order part of the product of the multiplicand times the multiplier.

DVDND	Is the location of the high-order part of the dividend.
DVSOR	Is the location of the high-order part of the divisor.
QTNT	Is the location of the high-order part of the quotient of the dividend divided by the divisor.
SPDVSR	Is the location of the single-precision divisor.
ERRLOC	Is the location to which control is to return if the divisor is zero. This should be set by an ASSIGN statement in FORTRAN and MADTRAN programs.

Restrictions: The tag of an operand may not be 4.

The operands may not be indirectly addressed.

Execution: DFAD causes the double-precision numbers to be added together. The result is double-precision.

DFSB causes one double-precision number to be subtracted from the other. The result is double-precision.

DFMP multiplies two double-precision numbers together. The result is double-precision.

DFDP divides one double-precision number by another. The result is double-precision.

SFDP divides a double-precision number by a single-precision number. The result is double-precision.

DCEXIT allows the user to specify the location to which control is to go if division by zero is attempted.

CTSS SUBPROGRAM

Entry Name:	DIM
Purpose:	To duplicate the FAP coding for the FORTRAN built-in function, DIMF, for use with MAD-coded subprograms.
Length:	7 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
X = DIM.(Y,Z)	CALL DIM,Y,Z
	STØ X

where:

the arguments and functions are floating-point.

Usage:	The routine finds the positive difference between the two arguments, i.e., Arg_1 -MIN (Arg_1, Arg_2).
Identification:	MDDIM appears in columns 73-77 of the binary deck and the symbolic deck.

CTSS SUBPROGRAM

Entry Name:	EXP Version II
Purpose:	Computes e^x for a single floating-point argument.
Length:	124 octal locations plus four temporary erasable locations.
Transfer Vector:	None
Error Procedure:	If the argument is greater than 88.028, the subprogram ERRØR is called. If the argument is less than -88.028, a result of zero is returned.

Calling Sequences:

FORTRAN	MAD	EAP
EX = EXPF(X)	EX = EXP.(X)	CLA X
		TSX $EXP,4
		STØ EX

where:

X is a floating-point number between -88.028 and +88.028.

References:	SHARE Distribution Nos. 507 and 571 IB FXP

CTSS SUBPROGRAM

Entry Name:	EXP(1
Purpose:	To compute I^J, where I and J are fixed-point variables.
Length:	45 octal locations plus two temporary locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequences:

EAP	FORTRAN
CLA I LDQ J TSX $EXP(1,4 STØ ITØJ	ITØJ = I**J

where:

I	is a fixed-point variable, stored in the decrement.
J	is a fixed-point variable, stored in the decrement.
ITØJ	is the fixed-point result, stored in the decrement.

CTSS SUBPROGRAM

Entry Name:	EXP(2 Version II
Purpose:	To compute X^K, where X is a floating-point variable and K is a fixed-point variable.
Length:	131 octal locations plus two temporary erasable locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If there is a large negative exponent and small base, the divide check light is turned on and the subprograms ERRØR and LDUMP are called.

Calling Sequences:

FORTRAN

```
Y = X ** K
```

FAP

```
CLA    X
LDQ    K
TSX    $EXP(2,4
STØ    Y
```

where:

X is a floating-point variable and

K is a fixed-point variable, stored in the decrement in FAP.

CTSS SUBPROGRAM

Entry Name:	EXP(3 Version II
Purpose:	To compute Y**Z, where Y and Z are floating-point variables.
Length:	236 octal locations plus four temporary erasable locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If there is a negative base and non-integral exponent, the subprograms ERRØR and LDUMP are called.

Calling Sequences:

```
          FORTRAN                    EAP

          W = Y ** Z                 CLA    Y
                                     LDQ    Z
                                     TSX    EXP(3,4
                                     STØ    W
```

where:

W, Y and Z are floating-point variables.

CTSS SUBPROGRAM

Entry Names:	FINT, MINT
Purpose:	To convert FORTRAN integers to MAD integers, or MAD integers to FORTRAN integers.
Length:	33 octal locations
Transfer Vector:	WRFLX
Error Procedure:	If a MAD integer is too large to be converted into a FORTRAN integer, the following message is printed: 'MAD INTEGER EXCEEDS 32767' and the MAD integer module 32768 is taken as the argument.

Calling Sequences:

```
          FORTRAN:   EQUIVALENCE (A,J)
                     A= FINT (I)            I = MINT (J)

          MAD:       J = FINT.(I)           I = MINT.(J)
                     INTEGER J,FINT.,I      INTEGER I,MINT.,J

          FAP:       TSX    $FINT,4         TSX    $MINT,4
                     PZE    I               PZE    J
                     STØ    J               STØ    I
```

where:

I	refers to a MAD (full word) integer.
J	refers to a FORTRAN (decrement) integer.
A	is equivalent to J.

Execution:

FINT converts a MAD integer into a FORTRAN integer. If the MAD integer is larger than 32767, a message is printed, and then the integer modulo 32768 is taken as the argument.

MINT converts a FORTRAN integer into a MAD integer.

<u>CTSS SUBPROGRAM</u>

Entry Name:	FLIP
Purpose:	To transpose a matrix.
Length:	132 octal locations
Transfer Vector:	.03311, EXIT
Calling Sequence:	MAD: EXECUTE FLIP.(NAME,M,N)

where:

NAME	is a two-dimensional array dimensioned as (MAX(M,N), MAX(M,N))
M	is an integer variable corresponding to the number of rows in the array NAME.
N	is an integer variable corresponding to the number of columns in the array NAME.

CTSS SUBPROGRAM

Entry Names:	INDV, DPNV
Purpose:	To obtain in floating-point arithmetic the numerical solution of a system of Nth order, non-linear, simultaneous ordinary differential equations, essentially by writing the initial conditions and differential equations in any desired FORTRAN or FAP language.
Length:	626 octal locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If sense light 1 is not on for the first entry to INDV or if there are more than 50 dependent variable statements, the subprogram ERRØR is called. Upon return to INDV, the subprogram LDUMP is called.

Calling Sequences:

FORTRAN

```
X = INDVF(X,H)
```

FAP

```
CLA    X
LDQ    J
TSX    $INDV,4
STØ    X
```

where:

X   is the independent variable and

H   is the increment.

FORTRAN

```
Y = DPNVF(Y,DY)
```

FAP

```
CLA    Y
LDQ    DY
TSX    $DPNV,4
STØ    Y
```

where:

Y   is the dependent variable and

DY   is the increment using Adams four point formula.

Reference: SHARE Distribution Nos. 413 and 827, GL AIDE1

CTSS SUBPROGRAM

Entry Name:	INT
Purpose:	To duplicate the FAP coding for the FORTRAN built-in function, INTF, for use with MAD-coded subprograms.
Length:	7 locations
Transfer Vector:	None

Calling Sequence:

```
MAD                 FAP

X = INT.(Y)         CALL   INT,Y
                    STØ    X
```

where:

the argument and function are floating-point.

Usage:	The routine truncates the argument (sign of argument times largest integer less than or equal to absolute value of the argument).
Identification:	MDINT appears in columns 73-77 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Names:	IØSET, IØPAR, IØEND, IØSCP, IØITR
Purpose:	Calls to these programs are compiled into the MAD program during MADTRAN translation of FORTRAN I/O statements involving iterations.
Length:	71 octal locations
Transfer Vector:	None
TIA to supervisor:	None
COMMON:	None
Error Procedure:	None

Calling Sequence:

```
          FORTRAN:   PRINT 1, (A(I), I = J,K,L)
```

where:

PRINT could be also READ, WRITE OUTPUT TAPE, READ INPUT TAPE.

1 is the format statement number.

A is the name of an array where the Ith element, I+Lth element, ... (until J+(N*L) is greater than K) will be printed.

```
          MAD:      PRINT FØRMAT ALPHA, L1, IØSET.(LØC1)
                    EXECUTE IØPAR.(L2)
                    EXECUTE IØEND.(LØC2)
                    EXECUTE IØITR.(V, M1, M2, M3)
                    EXECUTE IØSCP.
```

where:

PRINT FØRMAT could also be READ FORMAT, WRITE BCD TAPE or READ BCD TAPE.

ALPHA is the format name.

L1 and L2 are normal I/O lists.

LØC1 is the location where the MAD sequence begins.

LØC2 is the location where the program will go after completing the list. If omitted, control will return to the statement following 'EXECUTE IØEND.'

V is an iteration variable (usually used for indexing).

M1 is the initial value of the iterative variable V.

M2 is the last value of V.

M3 is the increment to be used on V.

Execution:

Executing IØSET will cause the program to leave the I/O list without terminating the format, transferring control to LØC1.

Executing IØPAR will cause the list L2 to be written (or read) as though part of the original I/O list. The format will be continued exactly as though the list were part of the original I/O list.

Executing IØEND terminates the I/O list and returns as indicated above.

Executing IØITR causes V to be set to its initial value, M1, and the values of M2 and M3 to be saved for use of IØSCP.

Executing IØSCP causes the value of V to be compared with M2, and, if less than M2, causes it to be incremented by M3 and then returns control to the beginning of the corresponding iteration; otherwise, control goes to the statement following the 'EXECUTE IØSCP.'

Restrictions:

When used in the list for IØPAR, multiply subscripted arrays must appear with multiple subscripts or with a variable single subscript (i.e., if BB is a multiply subscripted array, then either BB or BB(5) is illegal). IØITR and IØSCP may be nested three deep, but there must be an 'EXECUTE IØSCP.' corresponding to each use of IØITR for proper nesting.

CTSS SUBPROGRAM

Entry Name:	LØG
Purpose:	Computes the floating-point natural logarithm.
Length:	127 octal locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If the argument is less than or equal to zero, then control is transferred to the subprogram ERRØR. The subprogram LØG then transfers to LDUMP.

Calling Sequence:

FORTRAN	MAD	EAP
ELNX = LØGF(X)	ELNX = LØG.(X)	CLA X
		TSX $LØG,4
		STØ ELNX

where:

X	is a floating-point number greater than zero.
ELNX	is equal to $\log_e$ X, in floating-point.

Reference: SHARE Distribution No. 665, IB LØG 3

Examples:

```
    1)          FORTRAN:  PRINT 2, (A(I), I = 1,20)

                MAD:      PRINT ØNLINEFØRMAT QQ0002, IØSET.(QQ0003)
                 QQ0003   EXECUTE IØITR.(I, 1,20,1)
                          EXECUTE IØPAR.(A(I))
                          EXECUTE IØSCP.
                          EXECUTE IØEND.

    2)          FORTRAN:  PRINT2,((B(I,J), I = 1,3), J = 1,3)

                MAD:      PRINT ØNLINEFØRMAT QQ0002, IØSET.(QQ0005)
                 QQ0005   EXECUTE IØITR. (I, 1,3,1)
                          EXECUTE IØITR. (J, 1,3,1)
                          EXECUTE IØPAR. (B(I,J))
                          EXECUTE IØSCP.
                          EXECUTE IØSCP.
                          EXECUTE IØEND.
```

CTSS SUBPROGRAM

Entry Name:	MAX0
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, MAX0F, for use with MAD-coded subprograms.
Length:	25 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
J = MAX0.(I_1,....,I_n)	CALL MAX0,I_1,....,I_m
	STØ J

where:

the arguments (any number of arguments greater than one) are fixed-point and the function is floating-point.

Usage:	The routine finds the maximum value of the arguments.
Identification:	MDMAX0 appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Names:	MAX1, XMAX0
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, MAX1F and XMAX0F, for use with MAD-coded subprograms.
Length:	22 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
J = MAX1.(Y_1,...,Y_n)	CALL MAX1,Y_1,...,Y_n STØ X
J = XMAX0.(I_1,...,I_n)	CALL XMAX0,I_1,...,I_n STØ J

where:

the arguments and function of MAX1 are floating-point,

the arguments and function of XMAX0 are fixed-point,

and there can be any number (greater than one) of arguments.

Usage:	The routine finds the maximum value of the arguments.
Identification:	MDMAX1 appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	MINO
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, MINOF, for use with MAD-coded subprograms.
Length:	26 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
X = MINO.(I_1,...,I_n)	CALL MINO,I_1,...,I_n STØ X

where:

the arguments are fixed-point

and the function is floating-point.

Usage:	The function is used to find the smallest value of the set, I_1,...,I_n
Identification:	MDMINO appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Names: MIN1, XMINO

Purpose: To duplicate the FAP coding of the FORTRAN built-in functions, MIN1F and XMINOF, for use with MAD-coded subprograms.

Length: 23 octal locations

Transfer Vector: None

Calling Sequence:

MAD	FAP
X = MIN1.(Y_1,...,Y_n)	CALL MIN1,Y_1,...,Y_n STØ X
X = XMINO.(I_1,...,I_n)	CALL XMINO,I_1,...,I_n STØ X

where:

the arguments of MIN1 are floating-point and the mode of the function is floating-point,

the arguments of XMINO are fixed-point and the mode of the function is floating-point,

and for either entry the number of arguments is greater than 1.

Usage: The routines find the minimum value of at least two or more arguments.

Identification: MDMIN1 appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	MØD
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, MØDF, for use with MAD-coded subprograms.
Length:	14 octal locations
Transfer Vector:	None

Calling Sequence:

```
        MAD                   FAP

        Z = MØD.(X,Y)         CALL  MØD,X,Y
                              STØ   Z
```

where:

the arguments and function are floating-point.

Usage:	The function is defined as $ARG_1-(ARG_1-/ARG_2)*ARG_2$, where (X) = integral part of X.
Identification:	MDMØD appears in columns 73-77 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	RANNØ, SETU
Purpose:	Generates a floating-point number between 0 and 1.0 with rectangular distribution. The cycle time for each value of SETU is 2^{35}.
Length:	42 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:
for RANNØ-

```
     FORTRAN          MAD             EAP

     A=RANNØF(X)      A=RANNØ.(X)     TSX    $RANNØ,4
                                      STØ    A
```

where:

X Is a dummy argument.

A Is a floating-point random number generated by the formula

$$R_n = R_{n-1}(2^{17}+3)(\text{MØD}\,2^{35})$$

R_o =1 unless the subprogram SETU is used to change it.

for SETU-

```
     FORTRAN          MAD             EAP

     B=SETUF(I)       B=SETU.(I)      CLA    I
                                      TSX    $SETU,4
                                       .
                                       .
                                       .
                                  I    PZE    ,,5
```

where:

I Is a fixed-point variable used to change the starting value of R_o.

B Is a dummy argument.

CTSS SUBPROGRAM

Entry Name:	SIMCS
Purpose:	Simulates a FORTRAN function call to XSIMEQ. This routine is used by the library subprogram XSMEQ.
Length:	42 octal locations
Transfer Vector:	XSIMEQ
Common:	77775

Calling Sequence:

```
     MADTRN:   CALL SIMCS (MXFR,NFR,LFR,A,B,D,ARY,MFR)

     MAD:      EXECUTE SIMCS.(MXFR,NFR,LFR,A,B,D,ARY,MFR)
```

where:

MXFR	is a FORTRAN (decrement) integer giving the maximum number of rows the matrix A may have.
NFR	is a FORTRAN (decrement) integer giving the number of rows or columns in the matrix A at the time XSMEQ is called.
LFR	is a FORTRAN (decrement) integer giving the number of columns in matrix B.
A	is an unsubscripted floating-point variable referring to the square matrix A. Upon return, the answers (the X matrix) will replace the A matrix.
B	is an unsubscripted floating-point variable referring to the matrix B.
D	is a floating-point variable by whose value the value of the determinant of the matrix A will be scaled. Upon return, D = D*Det(A).
ARY	refers to a one-dimensional array whose length is greater than or equal to NFR.
MFR	upon return will contain a FORTRAN (decrement) integer which will be

1 if the solution was successful
2 if an overflow occurred
or 3 if the matrix A is singular.

Execution: Given the above calling sequence, SIMCS reforms it in such a way as to simulate a FORTRAN function call. XSIMEQ is called by this routine. Upon return from XSIMEQ, the resulting integer value is stored in MFR.

A programmer may call this routine instead of XSIMEQ if he first

1. alters his integers to make them FORTRAN (decrement) integers (by multiplying each of them by 2.P.18, or shifting the integers left 18 places, or calling FINT)

and 2. interchanges rows and columns in the A and B matrices so they appear as FORTRAN two-dimensional arrays.

and if upon return he

1. alters the resulting FORTRAN (decrement) integer to transform it into a MAD (address) integer (by dividing it by 2.P.18, or shifting the integer right 18 places, or calling MINT)

and 2. interchanges rows and columns of the X (or A) matrix so it appears as a MAD two-dimensional array.

Restrictions: All integers are FORTRAN (decrement) integers.

The matrix A must be a square matrix.

The matrices A and B must not be subscripted in the call. Further, their 'base elements' must be set to 1, that is, A(1) is the same as A(1,1).

Reference: CC-174

CTSS SUBPROGRAM

Entry Name:	SIN, CØS
Purpose:	Computes the sine or cosine of a floating-point radian argument.
Length:	172 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

```
     FORTRAN               MAD                EAP

     SINX = SINF(X)        SINX = SIN.(X)     CLA    X
                                              TSX    $SIN,4
                                              STØ    SINX
```

where:

X	is the angle in floating-point radians.
SINX	is the computed sine of X in floating-point.

Reference: SHARE Distribution No. 510, IB SIN 1

CTSS SUBPROGRAM

Entry Name:	(SLI) Version II
Purpose:	To provide list indexing for the input of nonsubscripted arrays.
Length:	17 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

```
        FORTRAN                              EAP

        DIMENSIØN SYMBØL (100)               TSX    $(SLI),4
        READ 1, SYMBØL                       PZE    SYMBØL + 1
                                             PZE    100
                                              .
                                              .
                                              .
                                             BSS    99
                                      SYMBØL BSS    1
```

where:

SYMBØL	is the first location in the array to be indexed.
100	is the number of variables in the array SYMBØL to be transmitted.

CTSS SUBPROGRAM

Entry Name:	(SLØ) Version II
Purpose:	To provide list indexing for the output of nonsubscripted arrays.
Length:	17 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

```
        FORTRAN                              EAP

        DIMENSIØN SYMBØL (100)               TSX    $(SLØ),4
        PRINT 1, SYMBØL                      PZE    SYMBØL + 1
                                             PZE    100
                                              .
                                              .
                                              .
                                             BSS    99
                                      SYMBØL BSS    1
```

where:

SYMBØL	is the first location in the array to be indexed.
100	is the number of variables in the array SYMBØL to be transmitted.

CTSS SUBPROGRAM

Entry Name:	SQRT, SQR
Purpose:	Computes the square root of a floating-point argument with either entry.
Length:	110 octal locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If the argument is negative, control transfers to the subprogram ERRØR and then SQRT transfers to LDUMP.

Calling Sequence:

```
     FORTRAN               MAD                 FAP

     SQX = SQRTF(X)        SQX = SQRT.(X)      CLA    X
                                               TSX    $SQRT, 4
                                               STØ    SQX
```

where:

X	is a floating-point variable whose square root is to be calculated.
SQX	is the square root of X in floating-point.

Reference: SHARE Distribution No. 703, CS SQT4

CTSS SUBPROGRAM

Entry Name:	TAN, CØT
Purpose:	Computes the tangent X or cotangent X for any single precision floating-point argument given in radians.
Length:	244 octal locations
Transfer Vector:	ERRØR, LDUMP
Error Procedure:	If the argument for TAN is greater than 2^{25} -1 or if the argument for CØT is less than 2^{126}, then control is transferred to the ERRØR subprogram. The subprogram TAN, CØT then calls LDUMP.

Calling Sequence:

FORTRAN	MAD	EAP
TANGX=TANF(X)	TANGX=TAN.(X)	CLA X
		TSX $TAN,4
		STØ TANGX

where:

X	is any single precision floating-point argument given in radians whose tangent is to be computed.
TANGX	is the floating-point result of the tangent X.

Reference: SHARE Distribution No. 507, IB TAN1

CTSS SUBPROGRAM

Entry Name:	TANH
Purpose:	Computes the hyperbolic tangent of X for any single precision floating-point argument given in radians.
Length:	143 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

```
     FORTRAN              MAD                  EAP

     ANS = TANHF(ARG)     ANS = TANH.(ARG)     CLA    ARG
                                               TSX    $TANH, 4
                                               STØ    ANS
```

where:

ARG	is a floating-point radian argument whose hyperbolic tangent is to be completed.
ANS	is the floating-point result of the hyperbolic tangent of ARG.

Reference: SHARE Distribution No. 507, IB TANH

CTSS SUBPROGRAM

Entry Name:	XDIM
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, XDIMF, for use with MAD-coded subprograms.
Length:	7 locations
Transfer Vector:	None

Calling Sequence:

```
        MAD                      FAP

        X = XDIM.(I,J)           CALL  XDIM,I,J
                                 STØ   X
```

where:

the arguments and function are fixed-point numbers.

Usage:	The routine finds the positive difference between the two arguments, i.e., ARG_1 - MIN(ARG_1, ARG_2).
Identification:	MDXDIM appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	XDTRM
Purpose:	To allow MAD and MADTRN routines to call XDETRM, a program to compute the value of a determinant. This value is then modified by a scale factor.
Length:	53 octal locations
Transfer Vector:	DETCS, FINT, MINT
Calling Sequence:	MADTRN: M = XDTRMF(N,NØ,A,D)
	MAD: M = XDTRM.(N,NØ,A,D) INTEGER M,N,NØ,XDTRM.

where:

N refers to an integer whose value is equal to the parameter N in the MADTRN statement 'DIMENSIØN A(N,N)' or is the third element in the MAD dimension vector describing the matrix A.

NØ refers to an integer giving the number of rows or columns in the matrix A at the time XDTRM is called.

A is an unsubscripted floating-point variable referring to the matrix. In a MAD program, the 'base element' of this array must be 1. Upon return, this matrix may be altered.

D is a floating-point variable by whose value the value of the determinant will be modified. Upon return, D = D*Det(A).

M upon return will be

1 if the solution was successful

2 if an overflow occurred

or 3 if the matrix A is singular.

Restrictions:

This program must not be used by FORTRAN routines.

All integers are expected to be normal MAD address integers.

The matrix A must be a square matrix with the 'base element' in the dimension vector defined as 1.

A in the calling sequence must not be subscripted.

<u>NOTE</u>: XDTRM. must appear in an integer declaration in a MAD program.

Reference: CC-174

CTSS SUBPROGRAM

Entry Names:	XINT, XFIX
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, XINTF and XFIXF, for use with MAD-coded subprograms.
Length:	12 octal locations
Transfer Vector:	None

Calling Sequence:

```
        MAD                     FAP

        J = XINT.(Y)            CALL   XINT,Y
                                STØ    J

        J = XFIX.(Y)            CALL   XFIX,Y
                                STØ    J
```

where:

the argument is a floating-point number and,

the function is fixed-point.

Usage:	The routine truncates the argument (sign of argument times largest integer less than or equal to absolute value of the argument).
Identification:	MDXINT appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	XLØC Version II
Purpose:	Finds the location where a variable is stored.
Length:	26 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

FORTRAN

```
L = XLØCF(N)
```

FAP

```
CLA    N
TSX    $XLØC,4
STØ    L
```

where:

N	is the variable whose location is to be found.
L	is the location of N.

CTSS SUBPROGRAM

Entry Name:	XMAX1
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, XMAX1F, for use with MAD-coded subprograms.
Length:	27 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
J = XMAX1.(Y_1,...,Y_n)	CALL XMAX1,Y_1,...,Y_n STØ J

where:

the arguments (any number of arguments greater than one) are floating-point and

the function is fixed-point.

Usage:	The routine finds the maximum value of the arguments.
Identification:	MDMX1 appears in columns 73-77 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	XMIN1
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, XMIN1F, for use with MAD-coded subprograms.
Length:	31 octal locations
Transfer Vector:	None

Calling Sequence:

MAD	FAP
J = XMIN1.(Y_1,...,Y_n)	CALL XMIN1,Y_1,...,Y_n
	STØ J

where:

the arguments are floating-point and

the function is fixed-point.

Usage:	The routines find the minimum value of the arguments.
Identification:	MDXMN1 appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	XMØD
Purpose:	To duplicate the FAP coding of the FORTRAN built-in function, XMØDF, for use with MAD-coded subprograms.
Length:	7 locations
Transfer Vector:	None

Calling Sequence:

```
        MAD                     FAP

        K = XMØD.(I,J)          CALL   XMØD,I,J
                                STØ    K
```

where:

the arguments and functions are fixed-point.

Usage:	The function is defined as $ARG_1-(ARG_1/ARG_2)$, where (X) = integral part of X.
Identification:	MDXMØD appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Names:	XSIGN, SIGN
Purpose:	To duplicate the FAP coding of the FORTRAN built-in functions, XSIGNF and SIGNF, for use with MAD-coded subprograms.
Length:	6 locations
Transfer Vector:	None

Calling Sequence:

```
        MAD                          FAP

        Z = SIGN.(X,Y)               CALL  SIGN,X,Y
                                     STØ   Z

        K = XSIGN.(I,J)              CALL  XSIGN,I,J
                                     STØ   K
```

where:

the argument and function of SIGN are floating-point and

the argument and function of XSIGN are fixed-point.

Usage:	The routine does a transfer of sign (Sign of Arg$_2$ times Arg$_1$).
Identification:	MDSIGN appears in columns 73-78 of the binary and symbolic decks.

CTSS SUBPROGRAM

Entry Name:	XSMEQ
Purpose:	To allow MAD and MADTRN routines to call XSIMEQ, a program to solve the matrix equation AX=B for the unknown matrix X.
Length:	116 octal locations
Transfer Vector:	FINT, MINT, SIMCS, FLIP

Calling Sequence:

```
     MADTRN:   M = XSMEQF(MXRØW,N,L,A,B,SCALE,ARRAY)

     MAD:      M = XSMEQ.(MXRØW,N,L,A,B,SCALE,ARRAY)
               INTEGER M,MXRØW,N,L,XSMEQ.
```

where:

MXRØW	refers to an integer whose value is equal to the parameter MXRØW of the MADTRN statement 'DIMENSIØN A(MXRØW,J)'.
N	is an integer giving the number of rows or columns in the matrix A at the time XSMEQ is called.
L	is an integer variable whose value is equal to the number of columns in matrix B.
A	is an unsubscripted floating-point variable referring to the matrix A. In a MAD program, the 'base element' of this array must be 1. Upon return, the answers (the X matrix) will replace the A matrix.
B	is an unsubscripted floating-point variable referring to the matrix B. In a MAD program, the 'base element' of this array must be 1. This matrix must be dimensioned in MADTRN as 'B(I,J)' where I and J are integer constants each greater than or equal to N.
SCALE	is a floating-point variable by whose value the value of the determinant of the matrix A will be scaled. Upon return, SCALE = SCALE * Det(A).

ARRAY	refers to a one-dimensional array whose length is greater than or equal to N.
M	upon return will be

1 if the solution was successful

2 if an overflow occurred

or 3 if the matrix A is singular.

Restrictions:

This program must not be called by FORTRAN routines.

All integers are expected to be normal MAD address integers.

The matrices A and B must not be subscripted in the call to XSMEQ. Their 'base elements' in their respective dimension vectors must be set to 1.

The matrix A must be a square matrix.

Each maximum subscript of matrix B must be greater than or equal to N, the number of rows of matrix A.

<u>NOTE</u>: XSMEQ. must appear in an integer declaration in a MAD program.

Reference: CC-174

CTSS SUBPROGRAM

Entry Name:	.01300
Purpose:	Computes Y^Z where Y and Z are floating-point variables.
Length:	106 octal locations
Transfer Vector:	SQRT, LØG, EXP, ERRØR, LDUMP
Error Procedure:	The subprogram ERRØR is called if $Y < 0$ and the Z is not an integer.

Calling Sequence:

MAD

```
X = Y.P.Z
```

EAP

```
CLA    Y
LDQ    Z
TSX    $.01300,4
STØ    X
```

where:

X, Y, and Z are floating-point variables.

CTSS SUBPROGRAM

Entry Name:	.01301
Purpose:	Computes X^K where X is a floating-point variable and K is a fixed-point variable.
Length:	43 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

MAD	FAP
Y = X.P.K	CLA X
	LDQ K
	TSX $.01301,4
	STØ Y

where:

X	is a floating-point variable.
K	is a fixed-point variable stored in the address in FAP.
Y	is a floating-point variable equal to X^K.

CTSS SUBPROGRAM

Entry Name:	.01311
Purpose:	To compute I^J where I and J are fixed-point variables.
Length:	42 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

MAD

```
ITØJ = I.P.J
```

FAP

```
CLA    I
LDQ    J
TSX    $.01311,4
STØ    ITØJ
```

where:

I, J, and ITØJ are fixed-point variables stored in the address in FAP.

CTSS SUBPROGRAM

Entry Name:	.03310, .03311
Purpose:	Computes the linear subscripts for arrays of two subscripts.
Length:	17 octal locations
Transfer Vector:	None
Error Procedure:	None

Calling Sequence:

MAD	EAP
A(I,J) = ...	CLA I LDQ J TSX $.03310,4 or .03311 TXH A,,ADIM STØ SUBSCR

where:

A	is the name of the array.
ADIM	is the location of the dimension vector.
I and J	are subscripts.
SUBSCR	is the linear subscript which is returned to the address of the AC.