1 SUMMARY

To **read a file held in Rutherford-Boeing format**. This file can contain either a sparse matrix or supplementary data. The HSL routines `MC54` and `MC55` can be used to write matrices and supplementary data in Rutherford-Boeing format, respectively. For information on the Rutherford-Boeing Sparse Matrix Collection (RBSMC), the user should consult the report

The Rutherford-Boeing Sparse Matrix Collection. I.S. Duff, R.G. Grimes, and J.G. Lewis. Report RAL-TR-97-031, Revision 1, 1999, Rutherford Appleton Laboratory.

ATTRIBUTES — Version: 1.0.0. **Types:** `MC56A`, `MC56AD` `MC56AI`, `MC56AC`, `MC56AZ`. **Remark:** This package is also included in the HSL Archive. **Language:** Fortran 77. **Original date:** September 2000. **Origin:** I.S. Duff, Rutherford Appleton Laboratory. **Licence:** A third-party licence for this package is available without charge.

2 HOW TO USE THE PACKAGE

2.1 The argument list

There is a single subroutine for both matrix data and supplementary files.

The single precision version

```
      CALL MC56A(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,NVEC,NE,
     *           LIP,IP,LIND,IND,LVALUE,VALUE,INFO)
```

The double precision version

```
      CALL MC56AD(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,NVEC,NE,
     *           LIP,IP,LIND,IND,LVALUE,VALUE,INFO)
```

The integer version

```
      CALL MC56AI(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,NVEC,NE,
     *           LIP,IP,LIND,IND,LVALUE,VALUE,INFO)
```

The complex version

```
      CALL MC56AC(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,NVEC,NE,
     *           LIP,IP,LIND,IND,LVALUE,VALUE,INFO)
```

The double precision complex version

```
      CALL MC56AZ(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,NVEC,NE,
     *           LIP,IP,LIND,IND,LVALUE,VALUE,INFO)
```

`ICNTL` is an `INTEGER` array of length `10` whose first component must be set by the user. It holds information on the input matrix and output file and controls the action of the subroutine. This argument is not altered by the subroutine.

`ICNTL(1)` must be set by the user to the unit number for the file on which the Rutherford-Boeing data is held.

`ICNTL(2)` to `ICNTL(10)` are currently not accessed by the routine.

`TITLE` is a `CHARACTER*72` variable that need not be set by the user. It will be set by the subroutine to the title line for the data.

`KEY` is a `CHARACTER*8` variable that need not be set by the user. It will be set by the subroutine to the identifying key for the matrix associated with the data being read.

CASEID is a CHARACTER*8 variable that need not be set by the user. It is not accessed if matrix data is being read. If a supplementary file is being read, CASEID will be set by the subroutine to the identifying key for the data. Different supplementary files of the same type for the same matrix will have different values for CASEID.

DATTYP is a CHARACTER*3 variable that need not be set by the user. It will be set by the subroutine to identify the type of matrix or supplementary data being read. For matrices it will have the format 'xyz' where:

x has the value r, c, i, p, or x.

y has the value s, u, h, z, or r.

z has the value a or e,

where the meanings are defined in the definition of the Rutherford-Boeing format for matrices in the report referenced above.

For supplementary data possible values for DATTYP are:

ord	... orderings
rhs	... right-hand sides
sln	... solutions
est	... estimates
evl	... eigenvalues
svl	... singular-values
evc	... eigenvectors
svc	... singular-vectors
sbv	... Schur basis vectors
sbm	... Schur basis matrix
sbp	... Schur basis parameters
ipt	... partition
icv	... covering
lvl	... Laplacian-values
lvc	... Laplacian-vectors
geo	... geometry
avl	... auxiliary values

POSITN is a CHARACTER*1 variable that need not be set by the user. It is not accessed if matrix data is being read. If a supplementary file is being read, it will be set by the subroutine to a blank unless DATTYP has a value equal to ord, rhs, sln, est, evc, svc, lvc, ipt, icv, or geo. Possible values are s, l, r corresponding to symmetric application or operation on the left or right. The exact meaning is dependent on the value of DATTYP (see documentation for RBSMC). The meaning of POSITN in each of these cases is as follows:

ord POSITN has the value s, l, or r according to whether the ordering is applied symmetrically, on the left, or on the right.

rhs, sln, est POSITN has the value r or l according to whether the linear system is of the form **Ax=b** or $\mathbf{A}^T\mathbf{x}=\mathbf{b}$ respectively.

evc, svc, lvc POSITN has the value s, l, or r according to whether the vectors represent both left and right eigensolutions (lvc and symmetric evc case), or the left or right vectors respectively.

ipt, icv POSITN has the value s, l, or r according to whether the partition or covering should be applied symmetrically, or on the left (rows) or right (columns) respectively.

geo POSITN has the value s, l, or r according to whether the geometry data applies symmetrically, or represents the left/row vertices or right/column vertices respectively.

ORGNIZ is a CHARACTER*1 variable that need not be set by the user. It is not accessed if matrix data is being read. If a supplementary file is being read and DATTYP is equal to rhs, it is set by the routine to indicate whether the right-hand side is sparse (s), dense (d), or elemental (e). Otherwise it is set to a blank character.

M is an INTEGER variable that need not be set by the user. It will be set by the subroutine. For matrix data, it will be set to the number of rows if the matrix is assembled (DATTYP(3:3) = a) and to the largest integer used as an index for an element variable if the matrix is in unassembled elemental form (DATTYP(3:3) = e). For supplementary data, the actual meaning depends on the value of DATTYP. For ord, rhs, sln, est, evc, svc, lvc, sbv, ipt, icv, avl it is the row dimension of the vectors involved; for evl, svl, lvl, sbm, sbp, geo it is the number of values, vectors, or points involved.

NVEC is an INTEGER variable that need not be set by the user. It will be set by the subroutine. For matrix data, it will be set to the number of columns if the matrix is assembled (DATTYP(3:3) = a) and to the number of elements if the matrix is in unassembled elemental form (DATTYP(3:3) = e). For supplementary data, NVEC will be set to the number of vectors in the supplementary file.

NE is an INTEGER variable that need not be set by the user. It will be set by the subroutine. For matrix data, it will be set to the number of entries in the matrix if the matrix is assembled (DATTYP(3:3) = a) and to the number of integer entries in the element information if the matrix is in unassembled elemental form (DATTYP(3:3) = e). For supplementary data, NE will be set to the number of entries in the supplementary file.

LIP is an INTEGER variable that must be set by the user to the length of the array IP. It is not altered by the subroutine.

IP is an INTEGER array of size LIP that need not be set by the user. For matrix data, it will be set to the position of the column starts in IND (DATTYP(3:3) = a) or to the start of the data for each element matrix in IND (DATTYP(3:3) = e). For supplementary data, if DATTYP is equal to rhs (ORGNIZ = s), ipt, or icv it will be set by the subroutine. For rhs (ORGNIZ = s), it will be set to the starts of each sparse right-hand side in IND and VALUE; for ipt and icv it will be set to the beginning of each partition or covering in IND.

LIND is an INTEGER variable that must be set by the user to the length of the array IND. It is not altered by the subroutine.

IND is an INTEGER array of length LIND that need not be set by the user. For matrix data, it will be set to either row indices by columns (DATTYP(3:3) = a) or lists of variables in each element (DATTYP(3:3) = e). For supplementary data, it will be set by the subroutine if DATTYP has the value ord, rhs (ORGNIZ = s), ipt, or icv. For ord, it will be set to the orderings; for rhs (ORGNIZ = s), it will be set to the row indices of the entries in the sparse right-hand sides; and for ipt and icv to the indices for each partitioning or covering.

LVALUE is an INTEGER variable that must be set by the user to the length of the array VALUE. It is not altered by the subroutine.

VALUE is a REAL (DOUBLE PRECISION in the D version, INTEGER in the I version, COMPLEX in the C version, or DOUBLE PRECISION COMPLEX in the Z version) array of length LVALUE that need not be set by the user. For matrix data, if reals are supplied, it will be set to the values of the matrix (DATTYP(3:3) = a) or element (DATTYP(3:3) = e) entries. For supplementary data, if DATTYP is not equal to ord, ipt, or icv, it will be set by the subroutine to the value of the numerical data.

INFO is an INTEGER array of length 5 that need not be set by the user. INFO(1) will be set to 0 in the event of a successful call to MC56A/AD/AI/AC/AZ and will have a negative value if an error is detected (see Section 2.2). In the case of an error, INFO(2) will hold further information. INFO(3) to INFO(5) are currently not used by the routine.

2.2 Error diagnostics

A successful return from MC56A/AD/AI/AC/AZ is indicated by a value of INFO(1) equal to zero. Possible nonzero values for INFO(1) are given below.

–1 Array IP too small to hold data being read. A suitable length is returned in INFO(2).

–2 Array IND too small to hold data being read. A suitable length is returned in INFO(2).

–3 Array VALUE too small to hold data being read. A suitable length is returned in INFO(2).

3 GENERAL INFORMATION

Workspace: None.

Use of common: None.

Other routines called directly: MC56 does not call any other subroutines.

Input/output: Only the Rutherford-Boeing file for the matrix or supplementary data that is read from unit ICNTL(1).

4 METHOD

The routine reads the header data first and, depending on the values, will read the rest of the data into the appropriate arrays.

5 EXAMPLES OF USE

We illustrate the use of the subroutine by a small program that reads a matrix from a file. The program then prints out the data read from the Rutherford-Boeing file.

```
C   This program reads a sparse matrix from the file on unit 5 (standard
C   input) and then prints it on unit 6.

      PROGRAM SPEC

      INTEGER          NZMAX,NMAX
      PARAMETER        (NZMAX=1000,NMAX=100)
      DOUBLE PRECISION VALUE(NZMAX)
      INTEGER          IND(2*NZMAX),IP(NMAX+1)
      INTEGER          ICNTL(10),M,N,NE,IW(NMAX),INFO(5),I,K,NVEC
      CHARACTER        TITLE*72,KEY*8,CASEID*8,DATTYP*3,POSITN*1,
     *                 ORGNIZ*1

      EXTERNAL         MC56AD

C Call MC56AD to read matrix data
      ICNTL(1) = 5
      CALL MC56AD(ICNTL,TITLE,KEY,CASEID,DATTYP,POSITN,ORGNIZ,M,N,NE,
     *            NMAX+1,IP,2*NZMAX,IND,NZMAX,VALUE,INFO)
      IF (INFO(1).LT.0) THEN
        WRITE(6,'(A,2I12)') 'Value of INFO is: ',INFO
        GO TO 500
      ENDIF

C Print matrix data
      WRITE(6,'(A/A,3I4)') 'Matrix data:','M,N,NE',M,N,NE
      WRITE(6,'(A/(20I4))') 'IP array:',(IP(I),I=1,N+1)
      WRITE(6,'(A/(20I4))') 'IND array:',(IND(I),I=1,NE)
      WRITE(6,'(A/(5D12.4))') 'VALUE array:',(VALUE(I),I=1,NE)

  500 STOP
      END
```

With the input:

```
Matrix for specification sheet example for MC56                         KASE_001
             3             1             1             1
rua                        3             3             5             0
(40I2)          (40I2)          (8E10.1E3)
 1 3 4 6
 1 2 2 2 3
  1.1E+000  2.1E+000  2.2E+000  2.3E+000  3.3E+000
```

this program produces, on unit 6:

```
Matrix data:
M,N,NE   3   3   5
IP array:
   1   3   4   6
IND array:
   1   2   2   2   3
VALUE array:
  0.1100D+01  0.2100D+01  0.2200D+01  0.2300D+01  0.3300D+01
```