BNL--39651

DE87 009146

GUIDE to
ACCELERATOR PHYSICS
PROGRAM SYNCH
— VAX VERSION 1987.2 —

[This is an on line manual in]

BNLDAG::DUA0:[PARSA1.SYNCH]

ZOHREH PARSA
&
ERNEST COURANT

BROOKHAVEN NATIONAL LABORATORY
ACCELERATOR DEVELOPMENT DEPARTMENT

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

MASTER

7300

GUIDE to
ACCELERATOR PHYSICS
PROGRAM SYNCH
VAX Version

ZOHREH PARSA
and
ERNEST COURANT

BROOKHAVEN NATIONAL LABORATORY

Associated Universities, INC.
Upton, New York 11973

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

This Guide is written to accommodate users of Accelerator Physics Data Base BNLDAG::DUA0:[PARSA1],

It describes the contents of the on line Accelerator Physics data base DUA0:[PARSA1.SYNCH]. SYNCH is a computer program used for the design and analysis of synchrotrons, storage rings and beamlines.

Table of contents:

AAA.CCC	Page 2	AGS.CCC	Page 2
ALTM1.CCC	Page 2	BCFD.CCC	Page 2
BETPTS.CCC	Page 2	BINFF.CCC	Page 2
BLANK.CCC	Page 2	BMI.CCC	Page 2
BMI1.CCC	Page 2	BPLTCOM.CCC	Page 2-3
BSTORE.CCC	Page 3	BWORK.CCC	Page 3
CDERIV.CCC	Page 3	CLC.CCC	Page 3
COMBLK.CCC	Page 3-6	CONST.CCC	Page 6
CONTRL.CCC	Page 6	COPY.CCC	Page 6
CORB.CCC	Page 6	CUT.CCC	Page 6
DIM.CCC	Page 7	FILES.CCC	Page 7
FLTN.CCC	Page 7	GRR.CCC	Page 7
IBSDAT.CCC	Page 7	INSTR.CCC	Page 7
KINET.CCC	Page 7	LBINFF.CCC	Page 7
LBSTORE.CCC	Page 7	MAGL.CCC	Page 7
MAINS.FOR	Page 7-13	MATCH.CCC	Page 13
MATCH2.CCC	Page 13	MATCH3.CCC	Page 13
MATCHD.CCC	Page 13	ME.CCC	Page 13
MG.CCC	Page 13	MI.CCC	Page 13
MINU.CCC	Page 13-14	MOVTOG.CCC	Page 14
OPLIST.CCC	Page 14	PAREXT.CCC	Page 14
PLTIN.CCC	Page 14	RANCOM.CCC	Page 14a
RDVECT.CCC	Page 14a	REVMAT.FOR	Page 14-17
STORE.CCC	Page 17	STR.CCC	Page 17
SUMCCC.CCC	Page 17-19	SVNAM.CCC	Page 19
SWTCH.CCC	Page 19-20	SYNI.FOR	Page 20-134
SYNII.FOR	Page 135-213	SYNNOPL.FOR	Page 213-365
SYNPL.FOR	Page 365-377	TRKINT.CCC	Page 377-378
TSW.CCC	Page 378	VAX.COM	Page 378
VERSION.CCC	Page 378	WRK.CCC	Page 378
XXYY.CCC	Page 378		

Welcome to DUA0:[PARSA1.SYNCH]

LOG

SEPTEMBER 1985	ZP	SET UP
OCTOBER 1985	ZP	UPDATE NEW VAX-VERSION
JANUARY 1987	ZP	NEW VAX-VERSION

GENERAL

Synch is a computer program for use in the design and analysis of synchrotrons, storage rings and beamlines. The program has modular organization, so the new capabilities can be added.

To prepare a VAX - version of "SYNCH":

Compile the following from BNLDAG::DUA0:[PARSA1.SYNCH.SYNVAX87]

MAINS.FOR (main program)

SYNI.FOR (subroutines with names from A to G)

SYNII.FOR (subroutines with names from G to P)

REVMAT.FOR (a recently corrected version of one subroutine)

SYNNOPL.FOR (subroutines with names from P to Z)

(alternate: SYNPL.FOR includes references to LBL DISSPLA system)

All the *.CCC files (COMMON files) must be present in the directory for compilation.

Then link resulting OBJ programs into EXE ***

\$ LINK/EXE=SYNCH MAINS,SYNI,SYNII,REVMAT,SYNNOPL

\$ RUN SYNCH.

To run "Synch": program looks for data in FOR002.Dat; main output is in FOR003.Dat; auxiliary output in various other files.

Our current executable file that is equivalent to all of this is SYNCH.EXE.

A current test case is SYNDEMO.DAT and its output is SYNDEM

You may run the vax version of SYNCH (with or without plotting)

interactively by creating the following SYNCH.COM file.

[Note that, you may add other DCL commands to the following example of SYNCH.COM file]

\$ ASSIGN SYNCH.DAT FOR002

\$ ASSIGN SYNCH.OUT FOR003

\$ RUN DUA0:[PARSA1.SYNCH]SYNCH.EXE

SYNCH.DAT contains the input statements starting with a RUN command and ending with a STOP statement, (see example of input file; SYNDEMO.DAT). The output from the RUN goes to SYNCH.OUT (for our example, SYNDEMO.OUT)

Having created the SYNCH.COM file, enter:

1. @SYNCH <return> [for interactive run]

or to batch the job enter:

2. \$ SUBMIT/NOPRINT SYNCH <return> [here you are
submitting SYNCH.COM file]

For information on how to use the CDC or IBM version of SYNCH see
the Guides to CDC and IBM versions of SYNCH respectively.

File: AAA.CCC;2

```
COMMON/AAA/BETA,ALPHA,AP,B,NB,R,X,RHO,NX,NV,XA,NA,NC,XX,A,AA,BB
DIMENSION BETA(2,84),ALPHA(84),AP(84),B(84),NB(1),R(84),X(84),
.RHO(252),NX(84),NV(84),XA(1),NA(1),NC(1),XX(84),A(84,84),DUM(7),
.AA(84,84),BB(84)
```

File: AGS.CCC;1

```
COMMON/AGS/ALPHA,BETA,RHO,BZ,AVAC,BMAX
```

C

File: ALTMI.CCC;2

C ALTMI.CCC

```
PARAMETER (NPMX=150)
COMMON/ALTMI/NPN,NP1(NPMX),NP2(NPMX)
```

File: BCFD.CCC;1

```
COMMON/BCFD/PAR(10),RMUX,RMUY,XGES,YGES,LAM
```

C

File: BETPTS.CCC;1

```
COMMON XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)
COMMON/BPLTCOM/MN,KW,BXX,BYX,NPLT
```

File: BINFF.CCC;2

```
COMMON /BINFF/ INFF(24,1000)
```

File: BLANK.CCC;1

```
COMMON STORE(17280),INFF(24,720),IWORK(2000)
```

File: BMI.CCC;1

```
COMMON /BMI/MI(5000)
COMMON/NELS/NELS
```

File: BMI1.CCC;1

```
COMMON/BMI1/MI1(16000)
COMMON/NELS1/NELS1
```

File: BPLTCOM.CCC;1

COMMON/BPLTCOM/MN,KW,BXX,BYX,NPLT

File: BSTORE.CCC;2

COMMON /BSTORE/ STORE(24000)

File: BWORK.CCC;1

COMMON /BWORK/ IWORK(2000)

File: CDERIV.CCC;4

COMMON/CDERIV/IFLAG,MR,NR,DP,NV(11),DAT(20),UZ,STOT,
1 DAT2(7),MP2FLG,NF,H0,LOCALPH,NUMALPH

File: CLC.CCC;1

COMMON/CLC/REG(109)

File: COMBLK.CCC;10

C AAA.CCC

COMMON/AAA/BETA,ALPHA,AP,B,NB,R,X,RHO,NX,NV,XA,NA,NC,XX,A,AA,BB
DIMENSION BETA(2,84),ALPHA(84),AP(84),B(84),NB(1),R(84),X(84),
.RHO(252),NX(84),NV(84),XA(1),NA(1),NC(1),XX(84),A(84,84),DUM(7),
.AA(84,84),BB(84)

C AGS.CCC

COMMON/AGS/ALPHA,BETA,RHO,BZ,AVAC,BMAX

C

C BCFD.CCC

COMMON/BCFD/PAR(10),RMUX,RMUY,XGES,YGES,LAM

C BETPTS.CCC

COMMON XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)

C BINFF.CCC

COMMON /BINFF/ INFF(24,1000)

C BMI.CCC

COMMON /BMI/MI(5000)

COMMON/NELS/NELS

C BMI1.CCC

COMMON/BMI1/MI1(16000)

COMMON/NELS1/NELS1

C BPLTCOM.CCC

COMMON/BPLTCOM/MN,KW,BXX,BYX,NPLT

C BSTORE.CCC

COMMON /BSTORE/ STORE(24000)

C BWORK.CCC

COMMON /BWORK/ IWORK(2000)

C CDERIV.CCC

COMMON/CDERIV/IFLAG,MR,NR,DP,NV(11),DAT(20),UZ,STOT,
1 DAT2(7),MP2FLG,NF,H0,LOCALPH,NUMALPH

C CLC.CCC

COMMON/CLC/REG(109)

C CONST.CCC

COMMON/CONST/PIO2,PI,TPI,PI2,RADEG

```

C CONTRL.CCC
COMMON/CONTRL/ERROR,MODE,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,
1      LDFLG,FIN
LOGICAL ERROR,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,LDFLG,FIN

C COPY.CCC
COMMON/COPY/CPYSW
LOGICAL CPYSW

C CORB.CCC
COMMON/CORB/TH(84),TV(84),NMON,NNH,NNV,IIH,IIV,
      .
      NAMONH,NAMONV,NACORH,NACORV

C DIM.CCC
COMMON /DIM/ LEND,MEND,ISAV,ISAV7,M7END,KADD,KADDR,MUNIT,MSYMP

C FILES.CCC
COMMON/FILES/JBIS,CYB0,TRK5,TRK6,ORB5,ORB12,IFILE,OFIL
INTEGER CYB0,TRK5,TRK6,ORB5,ORB12,OFIL

C FLTN.CCC
COMMON/FLTN/IFL(15)

C GRR.CCC
COMMON/GRR/IERR,BSW,VSW,RSW,QSW,XSW
LOGICAL IERR,BSW,VSW,RSW,QSW,XSW

C IBSDAT.CCC
COMMON/IBSDAT/ A(40),BTX(512),BTY(512),APX(512),APY(512),APP(512),
1      AP(512),AA(512),BB(512),CC(512),BPAR(10),SS(512)

C INSTR.CCC
COMMON/INSTR/OPNAME,NAME,OP,KA,KB,OBJA,OBJB,NXTM,TRA,LQ,NQ,LIN,
1      NIN,LFL,NFL,LBC,NBC,NTOT,IX,KIND,TYPE,ROWS,COLS,EXTR,
2      M,NF,NB,NI,MSUBR,ISIGN
INTEGER OPNAME,OP,OBJA,OBJB,TRA,TYPE,ROWS,COLS,EXTR

C KINET.CCC
COMMON/KINET/KNFLAG,TK,P,GAM,BETT,BETGAM,BRHO
LOGICAL KNFLAG

C MAGL.CCC
COMMON/MAGL/P(10),D,K

C MATCH.CCC
COMMON /MATCH/ KAT,KBT,MPAR(30),IPAR(30),MSR,MTR,LTRTAB,FNORM,
1      ICON(30),ITYPE(30),NPOS(30),IPOS(7,30),BET0(30),WT(30),
2      SIG(30),NAMBFC(42),ERPR,IVSAV,LVAR(30),WTVAR(30),VAL(30),
3      VARWTS,IPRPOS(6,30)
LOGICAL ERPR,LVAR,VARWTS

C MATCHD.CCC
DATA(NAMBFC(I),I=1,42)/4HNUX ,4HBX ,4HAX ,4HGX ,4HX ,
1      4HDX ,4HNUY ,4HBY ,4HAY ,4HGY ,4HY ,4HDY ,
2      4HS ,4HTHET ,
3      4HBXMX ,4HBYMX ,4HXXMAX ,4HQX ,4HQY ,4H ,4HGAMT ,
4      4HCHRX ,4HCHRY ,4HALPH ,4HYMAX ,4HBXMN ,4HBYMN ,4HXMN ,
5      4HYMIN ,4HCIRC ,4HTHTX ,4HTHTY ,4H ,4HMTRX ,4HRES ,
6      4HRESN ,4HBXBY ,4HRND2 ,4HRND ,4HWST ,4HDXD ,4HAXAY /

C MATCH2.CCC
COMMON/MATCH2/AA(30),BB(30),KBTS,SMINCAL,DWORD(10),
1      DWORD2(10),DWORD3(10),DWORD7(7,10),MINCOM
LOGICAL SMINCAL
INTEGER DWORD,DWORD2,DWORD3

C MATCH3.CCC
COMMON/MATCH3/NOPR,NONU,FLAG,IPR,IPP,MINFLG,MIFLG,MATFLG,GLOBAL
LOGICAL FLAG,NOPR,NONU,MIFLG,MATFLG,GLOBAL

```

```

C ME.CCC
COMMON /ME/ME(16000)

C MG.CCC
COMMON/MG/ML

C MI.CCC
COMMON /MI/MLL(3)

C ALTM.CCC
PARAMETER (NPMX=150)
COMMON/ALTM/ NPN,NP1(NPMX),NP2(NPMX)

C MINU.CCC
COMMON
1/MINERR/ ERP(30) ,ERN(30) D506 8
2/PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR D506 9
3/PAREXT/ U(30) ,NAM(30),NMIND(30) ,WERR(30) ,MAXEXT ,NU D506 10
4/LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET D506 11
5/VARIAN/ V(15,15) D506 12
7/FIX / IPFIX(15) ,XS(15) ,XTS(15) ,WTS(15) ,NPFIX D506 14
C/CASC/ JH, JL, Y(16) D506 19
F/DERIVA/ G(30) ,G2(30) D506 20
G/SIMVEC/ P(15,16) ,PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15) D506 21
J/VARIAT/ VT(15,15) D506 22
6/UNIT / ISYSRD ,ISYSWR ,ISYSPU D506 13
8/TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK D506 15
9/CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNM D506 16
A/CARD / CWORD ,CWORD2 ,CWORD3 ,WORD7(7) D506 17
B/MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA D506 18
K/COMMND/ KDUM ,CNAME(25) ,CNAM2(25) ,CNAM3(25),NNAME
INTEGER CNAME,CNAM2,CNAM3

C MOVTOG.CCC
COMMON/MOVTOG/TOG
LOGICAL TOG

C OPLIST.CCC
COMMON/OPLIST/LIST(200)

C PAREXT.CCC
COMMON /PAREXT/ U(30) ,NAM(30), NMIND(30), WERR(30), MAXEXT, NU

C PLTIN.CCC
COMMON/PLTIN/PLTIN
LOGICAL PLTIN

C RANCOM.CCC
COMMON /RANCOM/ IVAL

C STORE.CCC
COMMON/STORE/LMAX,LINF,LFILE,MAX,MIN,ICARD(11)
* LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF

C STR.CCC
COMMON/STR/GAM,A,LK,BK,F,LM,BL,LL,BQ,LTO,SMUD,CMUD
REAL LK,LM,LL,LTO

C SVNAM.CCC
COMMON/SVNAM/NAMRUN,BUF,BUFF,NRUNCOM(15)
CHARACTER*9 BUF
CHARACTER*8 BUFF

C SWITCH.CCC
COMMON/SWITCH/BEND,MSIZE,CYCSWT,VCSW,PV,MSSW,VPR,MHS,MODS,MINZER
LOGICAL CYCSWT,MSSW,VCSW,PV,VPR,MHS,MODS

C TRKINT.CCC
COMMON/TRKINT/G,EX,EY,OMSQ,SEND,VMX,VMN,DINT,BEG,DPR,JPR,

```



```

1   THET,PLT,NPT,NZ,PMAX,NSIZ,ITITLE(15)
    LOGICAL BEG,DPR,PLT
C TSW.CCC
    COMMON/TSW/TRSW,MCY(20),JM,LOCC
    LOGICAL TRSW
C VERSION.CCC
    COMMON/VERSION/CODE(2),ID,VER
C WRK.CCC
    COMMON/WRK/BZ,BQ,AZ,DPSI,SM,SL,PSI,WW,WT,BETW,P,Q,
1      T,U,BKZ,BKZP,CZ,SZ,CHZB,SHZB,PHI,BK,SA,BL,BM,
2      AQ,RHO,BETS,PSIWS,BKZI,BKZPI,G(6),AB,C,S,CHB,
3      SHB,E,EI,SZCB,SZBC,CZBC,CZCB,SZSB,SZBS,CZBS,CZSB,
4      ZW,ZT,BKI,Y,Z,Y1,Y2,Y3,Y4
C XYYY.CCC
    COMMON /XYYY/ XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)

```

File: CONST.CCC;1

```

COMMON/CONST/PIO2,PI,TPI,PI2,RADEG

```

File: CONTRL.CCC;1

```

COMMON/CONTRL/ERROR,MODE,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,
1      LDFLG,FIN
    LOGICAL ERROR,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,LDFLG,FIN

```

File: COPY.CCC;1

```

COMMON/COPY/CPYSW
    LOGICAL CPYSW

```

File: CORB.CCC;1

```

COMMON/CORB/TH(84),TV(84),NMON,NNH,NNV,IH,IIV,
    NAMONH,NAMONV,NACORH,NACORV

```

File: CUT.CCC;1

*** NOTE *** EACH OF THE FOLLOWING GROUPS SHOULD BE MADE INTO
SEPARATE FILES. THE NAME AND TYPE PRECEDE THE
COMMON BLOCK AND ARE NOTED BY C IN COLUMN 1.

THEN YOU HAVE TWO CHOICES.

1. CHANGE ALL THE REFERENCES IN SYNCH TO INCLUDE THEM BY
THEIR FILE NAMES, E.G. THE FIRST ONE BELOW TO BE IN A
FILE CALLED AAA.CCC AND THE FORTRAN CALL IN SYNCH TO
BE INCLUDE 'AAA.CCC'.
2. CREATE A TEXT FILE FROM THE *.CCC FILES. MY COMMAND
FILE IS AS FOLLOWS:

\$!

\$! CREATE A TEXT LIBRARY OF ALL THE INCLUDE FILES USED IN SYNCH

\$!

\$ LIBRARY/TEXT/CREATE LBL136:[ARDITH]SYNCHINC -

LBL136:[ARDITH]*.CCC

File: DIM.CCC;1

COMMON /DIM/ LEND,MEND,ISAV,ISAV7,M7END,KADD,KADDR,MUNIT,MSYMP

File: FILES.CCC;4

COMMON/FILES/JBIS,CYBO,TRK5,TRK6,ORB3,ORB5,ORB12,IFILE,OFIL
 INTEGER CYBO,TRK5,TRK6,ORB3,ORB5,ORB12,OFIL

File: FLTN.CCC;1

COMMON/FLTN/IFL(15)

File: GRR.CCC;1

COMMON/GRR/IERR,BSW,VSW,RSW,QSW,XSW
 LOGICAL IERR,BSW,VSW,RSW,QSW,XSW

File: IBSDAT.CCC;1

COMMON/IBSDAT/ A(40),BTX(512),BTY(512),APX(512),APY(512),APP(512),
 1 AP(512),AA(512),BB(512),CC(512),BPAR(10),SS(512)

C

File: INSTR.CCC;1

COMMON/INSTR/OPNAME,NAME,OP,KA,KB,OBJA,OBJB,NXTM,TRA,LQ,NQ,LIN,
 1 NIN,LFL,NFL,LBC,NBC,NTOT,IX,KIND,TYPE,ROWS,COLS,EXTR,
 2 M,NF,NB,NI,MSUBR,ISIGN
 INTEGER OPNAME,OP,OBJA,OBJB,TRA,TYPE,ROWS,COLS,EXTR

File: KINET.CCC;1

COMMON/KINET/KNFLAG,TK,P,GAM,BETT,BETGAM,BRHO
 LOGICAL KNFLAG

File: LBINFF.CCC;1

COMMON /BINFF/ INFF(24,1500)

File: LBSTORE.CCC;1

COMMON /BSTORE/ STORE(36000)

File: MAGL.CCC;1

COMMON/MAGL/P(10),D,K

File: MAINS.FOR;27

C*****MAIN*****
 IMPLICIT REAL*8 (A-H,O-Z)

C SYNCH— A COMPUTER SYSTEM FOR SYNCHROTRON DESIGN AND ORBIT
 * ANALYSIS. BY A.A.GARREN AND A.S.KENNEY, LAWRENCE BERKELEY LAB.

- PROGRAMMING QUESTIONS MAY BE DIRECTED TO ARDITH KENNEY, MATH
- AND COMPUTING, LBL. TELEPHONE X 5867.

C

```
INCLUDE 'CLC.CCC'
```

```
INCLUDE 'BSTORE.CCC'
```

```
INCLUDE 'BINFF.CCC'
```

```
INCLUDE 'BWORK.CCC'
```

```
INCLUDE 'XXYY.CCC'
```

```
INCLUDE 'BPLTCOM.CCC'
```

```
INCLUDE 'INSTR.CCC'
```

```
INCLUDE 'CONTRL.CCC'
```

```
INCLUDE 'STORE.CCC'
```

```
INCLUDE 'GRR.CCC'
```

```
INCLUDE 'DIM.CCC'
```

```
INCLUDE 'BMI.CCC'
```

```
INCLUDE 'BMI1.CCC'
```

```
INCLUDE 'MATCH3.CCC'
```

```
INCLUDE 'MATCH2.CCC'
```

```
INCLUDE 'FLTN.CCC'
```

```
INCLUDE 'SWTCH.CCC'
```

```
INCLUDE 'BCFD.CCC'
```

```
INCLUDE 'TRKINT.CCC'
```

```
INCLUDE 'TSW.CCC'
```

```
INCLUDE 'COPY.CCC'
```

```
INCLUDE 'OPLIST.CCC'
```

```
INCLUDE 'SVNAM.CCC'
```

```
INCLUDE 'CORB.CCC'
```

```
INCLUDE 'KINET.CCC'
```

```
INCLUDE 'CONST.CCC'
```

```
INCLUDE 'RANCOM.CCC'
```

```
INCLUDE 'FILES.CCC'
```

```
common/seed/nr
```

```
DIMENSION IC(24)
```

```
EQUIVALENCE (IC(1),OPNAME)
```

```
DIMENSION IW(6)
```

- TO ALTER STORAGE, CHANGE LMAX,MAX AND DIMENSION STORE
- TO INCREASE MATRIX SCRATCH SPACE, CHANGE ISAV AND ISAV7 FOR
- 3X3 OR 7X7 SIZES.

```
DATA LMAX,MAX,ISAV,ISAV7,KADD,KADDR/24000,1000,10,12,3,7/
```

```
DATA (IW(I),I=1,6) /0,0,0,-1,-1,-1/
```

```
DATA REG/109*0.0/
```

```
CALL SYSTM( 3,IW)
```

```
CALL SYSTM( 6,IW)
```

```
CALL SYSTM(13,IW)
```

```
CALL SYSTM(30,IW)
```

```
CALL SYSTM(36,IW)
```

```
CALL SYSTM(39,IW)
```

```
CALL SYSTM(42,IW)
```

```
nr=9999999
```

```
NPLT=0
```

```

DO 10 I=1,84
  TH(I) = 0.
10  TV(I) = 0.
C
  CALL INIT
  CALL SETOPS

9  FIN = .FALSE.
  PIO2 = DARCOS(0.D0)
  PI = 2.D0*PIO2
  TPI = 2.D0*PI
  PI2 = TPI
  RADEG = PI/180.D0
  MINFLG=1
  NONU = .FALSE.
  NOPR = .FALSE.
  MIFLG = .FALSE.
  MATFLG = .FALSE.
  GLOBAL = .FALSE.
  LFILE = 1
  LDFLG=.FALSE.
  KNFLAG = .FALSE.
  EMPTY = .FALSE.
  IERR=.FALSE.
  SMINCAL=.FALSE.
  CPYSW = .TRUE.
  MINCOM=3
  NOPR=.FALSE.
  NONU=.FALSE.
  LEND=LMAX
  MEND=MAX
  MIN=MAX
  M7END = MEND - ISAV
*      FILL BLANK PORTION OF ARRAY WITH ZEROES.
*  IFL IS USED MAINLY TO DISTINGUISH SYMBOLIC FL.PT. FROM REAL FL.PT.
2  CALL RDINST

4  CALL SWITCH
  IF (IERR) GO TO 6
  IF(ERROR) GO TO 6
  IF (FIN) GO TO 9
C  CHECK FOR REDEFINITION OF ELEMENT BY NAME
  IF (MODE.EQ.1.OR.MODE.EQ.2) CALL NAMCHK(M)
  IF(MODE.EQ.3) GO TO 3
  IF (.NOT.MSSW) GO TO 7
  CALL RDINF(IC,M)
  CALL MESH(M,KA,OPNAME)
  IF (MSSW) GO TO 3
C  IF PAGE OR COMMENT OUTSIDE SUBROUTINE, REUSE INFF SPACE.
7  IF ((OPNAME.EQ.4HPAGE.OR.OPNAME.EQ.3HREM).AND.(MODE.EQ.1)) GO TO 2
  MIN = MIN - 1
  NXTM=MIN
  CALL INFW( MIN,8,M )
  M=MIN
  GO TO 2

```

```

3  NXTM = INFF( 8,M )
   M=NXTM
   IF(.NOT.TRASW) GO TO 5
   M=TRA
   TRASW=.FALSE.
5  CALL RDINF( IC,M )
   GO TO 4
6  CALL ERRSR(M)
   GO TO 9
   END

```

C***** INIT *****

```

SUBROUTINE INIT
  IMPLICIT REAL*8 (A-H,O-Z)
  COMMON/MA/MLH(3)
  COMMON/MB/MLBND(3)
  COMMON/MC/MLCYAE(3)
  COMMON/MD/M3(3)
  COMMON/MG/MLCYX(3)
  COMMON/MH/MLFX(3)
  COMMON/MF/MLCYEM(3)
  COMMON/MI/MLMAGS(3)
  COMMON/MJ/MLMODQ(3)
  COMMON/MK/MLMOV(7)
  COMMON/ML/MLNPOL(3)
  COMMON/MM/MLREF(3)
  COMMON/MN/MLREV(3)
  COMMON/MO/MLROT(3)
  COMMON/MP/MLSXTP(3)
  COMMON/MQ/MLTRKCN(3)
  COMMON/MIM/MIM(100)
  COMMON/ME/ME(16000)
  INCLUDE 'MATCH2.CCC'
  INCLUDE 'BMI1.CCC'
  INCLUDE 'MINU.CCC'
  INTEGER DOLDWORD (3,3) /'PRIN','TOUT',' ','',
1      'MINI','MIZE',' ','','END ','RETU','RN ' /

```

DATA MIMAX/16000/

```

C
C  ZERO MI1 AND ME ARRAYS
  DO 10 I=1,MIMAX
    MI1(I) = 0
10  ME(I) = 0
C  DWORD IS IN MATCH2
  DO 20 I=1,3
    DWORD(I) = DOLDWORD(1,I)
    DWORD2(I) = DOLDWORD(2,I)
20  DWORD3(I) = DOLDWORD(3,I)
  DO 1 I=1,7
    DO 1 J=1,10
1  DWORD7(I,J)=1.D-8

```

```

  RETURN
  END

```

```

real function ran6(s)
real s
integer seed
data seed/99999999/
ran6=ran(seed)
return
end

```

```

real function ran6a(s)
integer s
integer seed
common/seed/nr
ran6a=ran(nr)
return
end

```

```

subroutine errtra
print*, ' ***** subroutine errtra ***** '
stop
end

```

```

integer function shftl(iword,num)
integer iword,num
shftl= ishft(iword,iabs(num))
return
end

```

```

integer function shftr(iword,num)
integer iword,num
shftr= ishft(iword,-iabs(num))
return
end

```

```

integer function land(iword1,iword2)
integer iword1,iword2
land=jiand(iword1,iword2)
return
end

```

```

integer function lor(iword1,iword2)
integer iword1,iword2
lor=jl原因(iword1,iword2)
return
end

```

```

real*8 function darcos(x)
real*8 x
darcos=dacos(x)
return
end

```

```

real*8 function dorsin(x)
real*8 x

```

```

darsin= dasin(x)
return
end

```

```

logical function tbit(i1,i2)
integer i1,i2
tbit= .false.
return
end

```

C***** BLOCK DATA *****

```

BLOCK DATA
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'CONTRL.CCC'

```

```

DATA ERROR /.FALSE./
INTEGER MODE / 1 /
INCLUDE 'MINU.CCC'
INCLUDE 'CDERIV.CCC'
INCLUDE 'MATCH.CCC'
INCLUDE 'MATCHD.CCC'
INCLUDE 'FLTN.CCC'
INCLUDE 'FILES.CCC'
INCLUDE 'VERSION.CCC'

```

C CNAME IS IN COMMND

```

DATA CNAME /      4HMINI ,4HSEEK ,4HSIMP ,4HMIGR ,4HMINO ,4HPUNC
7      ,4HPRIN ,4HEND ,4HFIX ,4HREST ,4HEXIT ,4HGRAD ,4HCALL
4      ,4HMATO ,4HCOVA ,4HIMPR ,4HERRO ,4HPAGE ,4HHESS ,4HCONT
1      ,4HSTAN ,4HRELE ,4H      ,4H      ,4HEND /
DATA CNAM2 /      4HMIZE ,4H      ,4HLEX ,4HAD ,4HS      ,4HH
7      ,4HTCUT ,4H      ,4H      ,4HORE ,4H      ,4HIENT ,4H FCN
4      ,4HUT ,4HRIAN ,4HOVE ,4HR DE ,4H      ,4HE      ,4HOUR
1      ,4HDARD ,4HASE ,4H      ,4H      ,4HRETU /
DATA CNAM3 /      2H      ,2H      ,2H      ,2H      ,2H      ,2H
7      ,2H      ,2H      ,2H      ,2H      ,2H      ,2H      ,2H
4      ,2H      ,2HCE ,2H      ,2HF      ,2H      ,2H      ,2H
1      ,2H      ,2H      ,2H      ,2H      ,2HRN /

```

DATA NNAME /25/

C IFL IS TABLE OF SPACE,+, -,., AND 0 -> 9.

```

DATA IFL /Z2D000000, Z31000000, Z32000000, Z33000000, Z34000000,
1      Z35000000, Z36000000, Z37000000, Z38000000, Z39000000,
2      Z30000000, Z2B000000, Z20000000, Z2E000000, Z2F000000/
DATA NV /1,2,3,4,5,6*0/

```

DATA ISYSRD,ISYSWR,ISYSPU/2,4,98/

DATA NBLOCK /0/

COMMON /INTC/ AREF,EMAX,SSSR,HFAC,SWAM,SWEX,IPMX

COMMON /INTP/ HPR,XX,EUB,ELB,SWIN,NEQS,IP,IT,NRKS

LOGICAL SWAM,SWEX,SWIN

INTEGER HFAC

REAL*16 HPR,XX

```

DATA IPMX,AREF,EMAX,SSSR,HFAC,SWAM,SWEX
1 /1024,1.D0,1.D-6,100.D0,2.,TRUE.,TRUE./
DATA MAXINT,MAXEXT /15,30/
DATA CYBO,TRK5,TRK6,ORB3,ORB5,ORB12,IFILE,OFIL /
x 20,21,22,27,23,24,25,26/
DATA CODE,ID,VER/4HVAX.,4HCDC.,1,4HB601/
END

```

File: MATCH.CCC;1

```

COMMON /MATCH/ KAT,KBT,MPAR(30),IPAR(30),MSR,MTR,LTRTAB,FNORM,
1 ICON(30),ITYPE(30),NPOS(30),IPOS(7,30),BET0(30),WT(30),
2 SIG(30),NAMBFC(42),ERPR,IVSAV,LVAR(30),WTVAR(30),VAL(30),
3 VARWTS,IPRPOS(6,30)
LOGICAL ERPR,LVAR,VARWTS

```

File: MATCH2.CCC;3

```

COMMON/MATCH2/AA(30),BB(30),KBTS,SMINCAL,DWORD(10),
1 DWORD2(10),DWORD3(10),DWORD7(7,10),MINCOM
LOGICAL SMINCAL
INTEGER DWORD,DWORD2,DWORD3

```

File: MATCH3.CCC;1

```

COMMON/MATCH3/NOPR,NONU,FLAG,IPR,IPP,MINFLG,MIFLG,MATFLG,GLOBAL
LOGICAL FLAG,NOPR,NONU,MIFLG,MATFLG,GLOBAL

```

File: MATCHD.CCC;2

```

DATA(NAMBFC(I),I=1,42)/4HNUX ,4HBX ,4HAX ,4HGX ,4HX ,
1 4HDX ,4HNUY ,4HBY ,4HAY ,4HGY ,4HY ,4HDY ,
2 4HS ,4HTHET ,
3 4HBXMX ,4HBYMX ,4HMAX ,4HQX ,4HQY ,4H ,4HGAMT ,
4 4HCHRX ,4HCHRY ,4HALPH ,4HYMAX ,4HBXMN ,4HBYMN ,4HXMIN ,
5 4HYMIN ,4HCIRC ,4HTHTX ,4HTHTY ,4H ,4HMTRX ,4HRES ,
6 4HRESN ,4HBXBY ,4HRND2 ,4HRND ,4HWST ,4HXDX ,4HAXAY /

```

File: ME.CCC;1

```

COMMON /ME/ME(16000)

```

File: MG.CCC;1

```

COMMON/MG/ML

```

File: MI.CCC;1

```

COMMON /MI/MLL(3)

```

File: MINU.CCC;8

```

COMMON
1/MINERR/ ERP(30) ,ERN(30) D506 8
2/PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR D506 9

```



```

3/PAREXT/ U(30) ,NAM(30),NMIND(30) ,WERR(30) ,MAXEXT ,NU      D506 10
4/LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET D506 11
5/VARIAN/ V(15,15)                                     D506 12
7/FIX / IPFIX(15) ,XS(15) ,XTS(15) ,WTS(15) ,NPFIX      D506 14
C/CASC/ JH, JL, Y(16)                                   D506 19
F/DERIVA/ G(30) ,G2(30)                                 D506 20
G/SIMVEC/ P(15,16) ,PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15) D506 21
J/VARIAT/ VT(15,15)                                     D506 22
6/UNIT / ISYSRD ,ISYSWR ,ISYSPU                         D506 13
8/TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK           D506 15
9/CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX      D506 16
A/CARD / CWORD ,CWORD2 ,CWORD3 ,WORD7(7)              D506 17
B/MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA               D506 18
K/COMMAND/ KDUIM ,CNAME(25) ,CNAM2(25) ,CNAM3(25) ,NNAME
      INTEGER CNAME,CNAM2,CNAM3

```

File: MOVTOG.CCC;1

C

```

COMMON/MOVTOG/TOG
LOGICAL TOG

```

C

File: OPLIST.CCC;1

```

COMMON/OPLIST/LIST(200)

```

File: PAREXT.CCC;1

```

COMMON /PAREXT/ U(30) ,NAM(30), NMIND(30), WERR(30), MAXEXT, NU

```

File: PLTIN.CCC;1

```

COMMON/PLTIN/PLTIN
LOGICAL PLTIN

```

File: REVMAT.FOR;5

```

C ..... REVMAT .....

```

```

SUBROUTINE REVMAT(N,M,V)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

C MAPS PARTICLE VECTOR V BY ELEMENT WITH INDEX M.
C ELEMENT MAY BE ANY LINEAR ELEMENT OR SEXTUPOLE.
C IF THE ELEMENT IS NOT COMPOSITE (E.G. AN MMM), LINEARIZED MATRICES
C THAT GIVE THE TRANSFORMATION RELATIVE TO V ARE CALCULATED AND
C STORED AT INDEX N.
C IF M IS A COMPOSITE ELEMENT, A NEW MATRIX IS NOT CALCULATED,
C THE OLD ONE IS STORED AT N.

C ELEMENTS DEFINED BY MAG AND SXTIP WILL BE RECALCULATED BY REVMAT,
C INCLUDING BR=BR*(1+DP/P).

C **NOTE** THIS ROUTINE USES MEND-3, MEND-4, AND MEND-5 (3X3 LOCS)

```

File: RANCOM.CCC

COMMON /RANCOM/ IVAL

File: RDVECT.CCC

COMMON /RDVECT/ ICOL, JCOL, KCOL, LCOL, NCOL,

+ NEXTRE

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'DIM.CCC'
INCLUDE 'CORB.CCC'
COMMON/MN/ML(3)
DIMENSION PAR(6),T(7,7),RW(3),V(1),V0(7),W(7),U(7)
EQUIVALENCE (ANG1,IANG1),(ANG2,IANG2),(BR,PAR(3))
INCLUDE 'CONST.CCC'

```

```

NOP = INFF(1,M)

```

```

C MOVE

```

```

IF (NOP.NE.4HMOVE) GO TO 100
CALL MOVE(M,N,V)
RETURN

```

```

C

```

```

100 DO 2 I=1,7
    U(I)=V(I)
2   V0(I)=V(I)

```

```

C SEXTUPOLE

```

```

IF (NOP.NE.4HSXTP) GO TO 1
CALL DATA(M,1,1,4,PAR)
INFF(4,N) = INFF(4,M)
INFF(5,N) = INFF(5,M)
CALL SXTP(N,V,PAR)
RETURN

```

```

C MAGNET DEFINED BY MAG INSTRUCTION

```

```

1   IF (NOP.NE.3HMAG) GO TO 7
    CALL DATA(M,1,1,6,PAR)
    GRAD=PAR(2)
    BZ=PAR(4)
    IRECT=0
    ANG1=PAR(5)
    ANG2=PAR(6)
    IF ( (IANG1.EQ.1H$).OR.(IANG2.EQ.1H$) ) IRECT=1
    IF (BZ.NE.0.) GO TO 111

```

```

C QUADRUPOLE

```

```

    CALL MODQ(M,N,V,W)
    GO TO 45
111 BR=BR*(1.D0+V(6))
    CALL MAGNET(N,PAR)
    CALL RTRV7(N,T,RW)
    CALL MXV7(T,V,W)

```

```

C BENDING MAGNET

```

```

    THETA = RW(2)
    IF (IRECT.EQ.1) GO TO 3
    E1=PAR(5)*RADEG
    E2=PAR(6)*RADEG
    GO TO 4

```

```

3   E1=.5D0*THETA
    E2=E1
4   E1T=E1+V(2)
    E2T=E2-W(2)
    X1=V(1)
    X2=W(1)
    IF (BZ.GE.0.) GO TO 40
    E1T=E1-V(2)
    E2T=E2+W(2)
    X1=-X1
    X2=-X2
40  PAR(5)=E1T/RADEG
    PAR(6)=E2T/RADEG
    D1=X1*DSIN(E1)/DCOS(E1T)
    D2=X2*DSIN(E2)/DCOS(E2T)
    DL=V(5)-W(5)
    PAR(1) = PAR(1) + DL - D1 - D2
    THETA = THETA*(1.D0+V(6)) + V(2) - W(2)
    RHO = PAR(1)/THETA
    PAR(4) = BR/RHO
    DO 5 I=1,3
5   ML(I) = MEND - 2 - I
    CALL DRIFT(MEND-3,D1)
    CALL MAGNET(MEND-4,PAR)
    CALL DRIFT(MEND-5,D2)
    CALL MMM(N,3,ML)
45  DO 50 I=1,5
50  V(I) = W(I)
    GO TO 9

```

C DRIFT

```

7   IF(NOP.NE.3HDRF) GO TO 8
    EL = FDAT(M,1)
    EL=EL*DSQRT(1.D0+V(2)*V(2))
    CALL DRIFT(N,EL)
    GO TO 10

```

C KICK

```

8   IF (NOP.NE.4HKICK) GO TO 82
    NAKICK = INFF(2,M)
    IF (NAKICK.NE.NACORH) GO TO 81
    TKH = TH(IIH)
    CALL STDAT(M,1,3,1,TKH)
    CALL KICK(M)
    IIH = IIH + 1
    IF (IIH.GT.NNH) IIH = 1
    GO TO 10
81  IF (NAKICK.NE.NACORV) GO TO 10
    TKV = TV(IIV)
    CALL STDAT(M,1,3,1,TKV)
    CALL KICK(M)
    IIV = IIV + 1
    IF (IIV.GT.NNV) IIV=1
    GO TO 10

```

C OTHER LINEAR ELEMENTS—MMM,ETC.

```

82  CONTINUE
10  CALL RTRV7(M,T,RW)
    CALL MXV7(T,V,V)
    CALL STOR7(N,T,RW)
    RETURN

```

```

9   CALL RTRV7(N,T,RW)
91  CALL MXV7(T,V0,W)
    DO 12 I=1,5
12  T(I,7)=V(I)-W(I)
    CALL STOR7(N,T,RW)

    RETURN
    END

```

File: STORE.CCC;1

```

COMMON/STORE/LMAX,LINF,LFILE,MAX,MIN,ICARD(11)
• LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF

```

File: STR.CCC;1

```

COMMON/STR/GAM,A,LK,BK,F,LM,BL,LL,BQ,LTO,SMUD,CMUD
REAL LK,LM,LL,LTO

```

File: SUMCCC.CCC;1

```

COMMON/AAA/BETA,ALPHA,AP,B,NB,R,X,RHO,NX,NV,XA,NA,NC,XX,A,AA,BB
DIMENSION BETA(2,84),ALPHA(84),AP(84),B(84),NB(1),R(84),X(84),
RHO(252),NX(84),NV(84),XA(1),NA(1),NC(1),XX(84),A(84,84),DUM(7),
AA(84,84),BB(84)

```

```

COMMON/AGS/ALPHA,BETA,RHO,BZ,AVAC,BMAX

```

```

COMMON /BCFD/ PAR(10),RMUX,RMUY,XGES,YGES,LAM

```

```

COMMON XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)

```

```

COMMON/BPLTCOM/MN,KW,BXX,BYX,NPLT

```

```

COMMON /BINFF/ INFF(24,720)

```

```

COMMON/BMI/MI(5000)

```

```

COMMON/NELS/NELS

```

```

COMMON/BMI1/MI1(16000)

```

```

COMMON/NELS1/NELS1

```

```

COMMON/BPLTCOM/MN,KW,BXX,BYX,NPLT

```

```

COMMON /BSTORE/ STORE(17280)

```

```

COMMON /BWORK/ IWORK(2000)

```

```
COMMON/CDERIV/IFLAG,MR,NR,DP,NV(11),DAT(20),UZ,STOT,
1 DAT2(7),MP2FLG,NF,H0,LOCALPH,NUMALPH
```

```
COMMON/CLC/REG(109)
```

```
COMMON/CONST/PIO2,PI,TPI,PI2,RADEG
```

```
COMMON/CONTRL/ERROR,MODE,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,
1 LDFLG,FIN
LOGICAL ERROR,RSRV,STOR,XEQ,TRASW,NFSW,EMPTY,INDEF,LDFLG,FIN
```

```
COMMON/COPY/CPYSW
LOGICAL CPYSW
```

```
COMMON/CORB/TH(84),TV(84),NMON,NNH,NNV,IIH,IIIV,
1 NAMONH,MANONV,NACORH,NACORV
```

```
COMMON/DIM/LEND,MEND,ISAV,ISAV7,M7END,KADD,KADDR,MUNIT,MSYML
```

```
COMMON/FLTIN/IFL(15)
```

```
COMMON/GRR/IERR,BSW,VSW,RSW,QSW,XSW
LOGICAL IERR,BSW,VSW,RSW,QSW,XSW
```

```
COMMON/IBSDAT/ A(40),BTX(512),BTY(512),APX(512),APY(512),APP(512),
1 AP(512),AA(512),BB(512),CC(512),BPAR(10),SS(512)
```

```
COMMON/INSTR/OPNAME,NAME,OP,KA,KB,OBJA,OBJB,NXTM,TRA,LQ,NQ,LIN,
1 NIN,LFL,NFL,LBC,NBC,NTOT,IX,KIND,TYPE,ROWS,COLS,EXTR,
2 M,NF,NB,NI,MSUBR,ISIGN
INTEGER OPNAME,OP,OBJA,OBJB,TRA,TYPE,ROWS,COLS,EXTR
```

```
COMMON/KINET/KNFLAG,TK,P,GAM,BETT,BETGAM,BRHO
LOGICAL KNFLAG
```

```
COMMON/MAGL/P(10),D,K
```

```
COMMON /MATCH/ KAT,KBT,MPAR(30),IPAR(30),MSR,MTR,LTRTAB,FNORM,
1 ICON(30),ITYPE(30),NPOS(30),IPOS(7,30),BET0(30),WT(30),
2 SIG(30),NAMBFC(42),ERPR,IVSAV,LVAR(30),WTVAR(30),VAL(30),
3 VARWTS,IPRPOS(6,30)
LOGICAL ERPR,LVAR,VARWTS
```

```
COMMON/MATCH2/AA(30),BB(30),KBTS,SMINCAL,DWORD(10),
1 DWORD2(10),DWORD3(10),DWORD7(7,10),MINCOM
LOGICAL SMINCAL
```

```
COMMON/MATCH3/NOPR,NONU,FLAG,IPR,IPP,MINFLG,MIFLG,MATFLG,GLOBAL
LOGICAL FLAG,NOPR,NONU,MIFLG,MATFLG,GLOBAL
```

```
DATA(NAMBFC(I),I=1,42)/5HNUX ,5HBX ,5HAX ,5HGX ,5HX ,
1 5HDX ,5HNUY ,5HBY ,5HAY ,5HGY ,5HY ,5HDY ,
2 5HS ,5HTHET ,
3 5HBXMX ,5HBYMX ,5HXXMAX ,5HQX ,5HQY ,5H ,5HGAMT ,
```

```

4      5HCHRX ,5HCHRY ,5HALPH ,5HYMAX ,5HBXMN ,5HBYMN ,5HXMIN ,
5      5HYMIN ,5HCIRC ,5HTHTX ,5HTHTY ,5H      ,5HMTRX ,5HRES ,
6      5HRESN ,5HBXBY ,5HRND2 ,5HRND ,5HWST ,5HDX ,5HAXAY /

```

```
COMMON /ME/ME(16000)
```

```
COMMON/MG/ML
```

```
COMMON /MI/MLL(3)
```

C

```
COMMON/MOVTOG/TOG
LOGICAL TOG
```

```
COMMON/OPLIST/LIST(200)
```

```
COMMON /PAREXT/ U(30) ,NAM(30) , NMIND(30) , WERR(30) , MAXEXT , NU
```

```
COMMON/PLTIN/PLTIN
LOGICAL PLTIN
```

```
COMMON /RANCOM/ IVAL
```

```
COMMON/STORE/LMAX,LINF,LFILE,MAX,MIN,ICARD(20)
```

• LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF

```
COMMON/STR/GAM,A,LK,BK,F,L,BL,LL,BQ,LTO,SMUD,CMUD
REAL LK,LM,LL,LTO
```

```
COMMON/SVNAM/NAMRUN
```

```
COMMON/SWCH/BEND,MSIZE,CYCSWT,VCSW,PV,MSSW,VPR,MHS,MODS
LOGICAL CYCSWT,MSSW,VCSW,PV,VPR,MHS,MODS
```

```
COMMON/TRKINT/G,EX,EY,OMSQ,SEND,VMX,VMN,DINT,BEG,DPR,JPR,
1  THET,PLT,NPT,NZ,PMAX,NSIZ,ITITLE(6)
LOGICAL BEG,DPR,PLT
```

```
COMMON/TSW/TRSW,MCY(20),JM,LOCC
LOGICAL TRSW
```

```
COMMON/WRK/BZ,BQ,AZ,DPSI,SM,SL,PSI,WW,WT,BETW,P,Q,
1  T,U,BKZ,BKZP,CZ,SZ,CHZB,SHZB,PHI,BK,SA,BL,BM,
2  AQ,RHO,BETS,PSIWS,BKZI,BKZPI,G(6),AB,C,S,CHB,
3  SHB,E,EI,SZCB,SZBC,CZCB,CZSB,SZSB,SZBS,CZBS,CZSB,
4  ZW,ZT,BKI,Y,Z,Y1,Y2,Y3,Y4
```

```
COMMON /XXYY/ XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)
```

File: SVNAM.CCC;2

```
COMMON/SVNAM/NAMRUN,BUF,BUFF,NRUNCOM(15)
CHARACTER*9 BUF
CHARACTER*8 BUFF
```

File: SWCH.CCC;1

COMMON/SWCH/BEND,MSIZE,CYCSWT,VCSW,PV,MSSW,VPR,MHS,MODS,MINZER
 LOGICAL CYCSWT,MSSW,VCSW,PV,VPR,MHS,MODS

File: SYNI.FOR;59

C ***** AFD *****

SUBROUTINE AFD(M,J,DELTA)
 IMPLICIT REAL*8 (A-H,O-Z)

C INCREMENTS BY DELTA THE J-TH FL PT PARAMETER OF ELEMENT M AND
 C RETURNS NEW VALUE TO STORAGE.

F=FDAT(M,J)
 F=F+DELTA
 CALL REPFLT(M,J,F)
 RETURN
 END

C ***** ALTC *****

SUBROUTINE ALTC(M,KA,P1)
 IMPLICIT REAL*8 (A-H,O-Z)
 INCLUDE 'GRR.CCC'
 INCLUDE 'SWCH.CCC'
 INCLUDE 'MAGL.CCC'

DIMENSION P1(1)
 INCLUDE 'CONST.CCC'

C DESIGN A SYMMETRIC CELL TO OBTAIN A SPECIFIED MUX VALUE

INTEGER AUXC/1/
 DO 100 I=1,8
 100 P(I)=P1(I)
 K=KA
 D=DCOS(TPI*P(8))
 CYCSWT=.TRUE.
 GUESS=P(K)
 CALL GRT (1,GUESS,1,AUXC)
 IF(IERR) RETURN
 CALL HED
 NAM=MNAME(M)
 WRITE(3,1000) NAM,(P(I),I=1,6)
 CALL HED
 CYCSWT=.FALSE.

C STORE THE FINAL VALUE OF THE ALTERED PARAMETER AS DATA FOR CELL

CALL REPFLT(M,K,P(K))
 CALL CELL(M,P)
 1000 FORMAT (//,19H NEW VALUES OF ALTC,2X,A5,11H PARAMETERS,/,3X,
 1 6(3X,F12.6))
 RETURN
 END

C ***** ARGINP *****

SUBROUTINE ARGINP(N1,I1,N2,I2,N3,I3,N4,I4,N5,I5,N6,I6,N7,I7,N8,I8,
 1N9,I9,N10,I10,ND,I,NSETS)
 IMPLICIT REAL*8 (A-H,O-Z)
 DIMENSION ND(10),I(10)
 NSETS=0


```

      IF (N1.EQ.0) GO TO 8
      NSETS=1
      ND(1)=N1
      I(1)=I1
      IF (N2.EQ.0) GO TO 8
      NSETS=2
      ND(2)=N2
      I(2)=I2
      IF (N3.EQ.0) GO TO 8
      NSETS=3
      ND(3)=N3
      I(3)=I3
      IF (N4.EQ.0) GO TO 8
      NSETS=4
      ND(4)=N4
      I(4)=I4
      IF (N5.EQ.0) GO TO 8
      NSETS=5
      ND(5)=N5
      I(5)=I5
      IF (N6.EQ.0) GO TO 8
      NSETS=6
      ND(6)=N6
      I(6)=I6
      IF (N7.EQ.0) GO TO 8
      NSETS=7
      ND(7)=N7
      I(7)=I7
      IF (N8.EQ.0) GO TO 8
      NSETS=8
      ND(8)=N8
      I(8)=I8
      IF (N9.EQ.0) GO TO 8
      NSETS=9
      ND(9)=N9
      I(9)=I9
      IF (N10.EQ.0) GO TO 8
      NSETS=10
      ND(10)=N10
      I(10)=I10
8     CONTINUE
      RETURN
      END

```

```

C ***** AROP *****
      SUBROUTINE AROP(M,INAM,IOP)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      INTEGER ELNUM
      COMPLEX CA,CB,CC
      LOGICAL PLUS,MINUS,MULT,DIV,POWER
      DIMENSION T(49),T1(49)
      DIMENSION RX(2,3),RY(2,3),RW(3),MI(2),RX1(2,3),RY1(2,3),RW1(3)
      DIMENSION INAM(4),NAM(4),PAR(4)
      EQUIVALENCE (NAM(1),PAR(1)),(CA,PAR(1))
      EQUIVALENCE (CB,PAR(3)),(A,PAR(1))

```

```

EQUIVALENCE (B,PAR(2))
EQUIVALENCE (T(1),RX(1,1)),(T(7),RY(1,1))
EQUIVALENCE (T1(1),RX1(1,1)),(T1(7),RY1(1,1))
EQUIVALENCE (CA,DA),(CB,DB),(CC,DC)
INTEGER doIPLUS/1H+/,doIMIN/1H-/,doIAST/1H*/,doISLSH/1H//,doIDAST/
*      2H**/,
1doISCAL/4HSCAL/,doISNGL/4HSNGL/,doIMXY/3HMX/,doIDBL/3HDBL/
DO 1 I=1,4
1  NAM(I) = INAM(I)
   KN = INFF(20,M)
   ITYP = INFF(21,M)
   PLUS = .FALSE.
   MINUS = .FALSE.
   MULT = .FALSE.
   DIV = .FALSE.
   POWER = .FALSE.
   IF (IOP.EQ.doIPLUS) PLUS = .TRUE.
   IF (IOP.EQ.doIMIN) MINUS = .TRUE.
   IF (IOP.EQ.doIAST) MULT = .TRUE.
   IF (IOP.EQ.doISLSH) DIV = .TRUE.
   IF (IOP.EQ.doIDAST) POWER = .TRUE.
   IF (KN.NE.doISCAL) GO TO 10
C  SCALAR OPERATION
   IF (ITYP.NE.doISNGL) GO TO 6
   IF (.NOT.MINUS) GO TO 5
   B = -B
5  IF (MINUS.OR.PLUS) C = A + B
   IF (MULT) C = A*B
   IF (DIV) C = A/B
   IF (POWER) C = A**B
   CALL STDAT(M,5,1,1,C)
   RETURN
6  IF (ITYP .EQ. doIDBL) GO TO 8
C  COMPLEX NUMBERS
   IF (.NOT.MINUS) GO TO 7
   CB = - CB
7  IF (MINUS.OR.PLUS) CC=CA+CB
   IF (MULT) CC = CA*CB
   IF (DIV) CC = CA/CB
11 CALL STDAT(M,5,1,2,CC)
   RETURN
8  IF ( .NOT. MINUS) GO TO 9
   DB=-DB
9  IF ( MINUS .OR. PLUS) DC=DA+DB
   IF (MULT) DC=DA*DB
   IF (DIV) DC=DA/DB
   GO TO 11
C  OPERATE ON MATRICES.  CONSIDER ONLY +,-,* FOR PRESENT
10 MA = ELNUM(INAM(1))
   MB = ELNUM(INAM(2))
   IF (.NOT.MULT) GO TO 15
   MI(1) = MA
   MI(2)=MB
   CALL MMM(M,2,MI)
   RETURN

```

```

15  IF (KN.NE.doIMXY) GO TO 30
    CALL RXY(MA,RX,RY,RW)
    CALL RXY(MB,RX1,RY1,RW1)
    IF (.NOT.MINUS) GO TO 20
    DO 17 J=1,3
    DO 16 I=1,2
    RX1(I,J) = -RX1(I,J)
    RY1(I,J) = - RY1(I,J)
16  CONTINUE
    RW1(J) = -RW1(J)
17  CONTINUE
20  DO 25 J=1,3
    DO 22 I=1,2
    RX(I,J) = RX(I,J) + RX1(I,J)
    RY(I,J) = RY(I,J) + RY1(I,J)
22  CONTINUE
    RW(J) = RW(J) + RW1(J)
25  CONTINUE
    CALL STXY(M,RX,RY,RW)
    RETURN
30  CALL RTRV7 (MA,T,RW)
    CALL RTRV7 (MB,T1,RW1)
    IF (.NOT.MINUS) GO TO 35
    DO 31 I=1,49
31  T1(I)= - T1(I)
    DO 32 I=1,3
32  RW1(I) = -RW1(I)
35  DO 37 I=1,49
37  T(I) = T(I) + T1(I)
    DO 38 I=1,3
38  RW(I) = RW(I) + RW1(I)
    CALL STOR7 (M,T,RW)
    RETURN
    END

```

```

C ***** AUXC *****
      SUBROUTINE AUXC(X,FMUX)
        IMPLICIT REAL*8 (A-H,O-Z)

```

```

C  FUNCTION EVALUATION FOR CELL DESIGN
      INCLUDE 'INSTR.CCC'

```

```

      INCLUDE 'MAGL.CCC'

```

```

      DIMENSION RX(2,3),RY(2,3),RW(3)
      EQUIVALENCE (D,FMU)
      P(K)=X
      CALL CELL(M,P)
      CALL RXY(M,RX,RY,RW)
      COSMUX = (RX(1,1) + RX(2,2)) * 0.5
      FMUX = COSMUX -FMU
      RETURN
      END

```

```

C ***** BDAT *****
      INTEGER FUNCTION BDAT(M,J)
      IMPLICIT INTEGER (A-Z)

```

```

C  RETRIEVES THE J-TH PIECE OF BCD INPUT DATA OF ELEMENT M

```

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
LOCI=INFF(16,M)*2-1
  LOC=LOCI+J-1
  ITEMP=ISTORE( LOC)
  BDAT=ITEMP
  RETURN
END

```

```

C ***** BEND *****
  SUBROUTINE BEND(M,V,PAR,IOP)

```

```

C 3X3 MATRICES FOR RECTANGULAR,ZERO-GRADIENT MAGNET, MODIFIED TO ACT
C ON OFF-MOMENTUM ORBITS. TRANSFORMATION IS RELATIVE TO A RAY V,
C GIVEN IN THE ORIGINAL COORDINATES, WHICH WILL BE TRACKED THROUGH
C THE MAGNET. THE RESULTING MATRIX IS TO BE USED BY CYX TO FIND
C BETATRON FUNCTIONS RELATIVE TO A NEW CLOSED ORBIT.
C IF IOP = 1, ACTS LIKE A MAG FOR THE SAME PARAMETERS.

```

```

C.....
C B      BEND          L          0.          BRHO      BZ          doI
C.....

```

```

  IMPLICIT REAL*8(A-H,O-Z)
  INCLUDE 'DIM.CCC'
  DIMENSION V(1),PAR(1),TAR(6),ML(3),T(49),RW(3)
  COMPLEX*16 R(6),Z,XI
  REAL*8 L
  INCLUDE 'CONST.CCC'

```

```

  XI = (0.,1.)
  IF (IOP.EQ.1) GO TO 2
  DEGRAD = 90.D0/PI02

```

```

  S0 = PAR(1)
  BR0 = PAR(3)
  B0 = PAR(4)
  R0 = DABS(BR0/B0)
  PHI0 = S0 / (2.*R0)
  SNPHI0 = DSIN(PHI0)
  L = 2. * R0 * SNPHI0

```

```

  X0 = V(1)
  AL0 = V(2)
  IF (B0.GT.0.) GO TO 3
  X0=-X0
  AL0=-AL0

```

```

3  CONTINUE
  DPP = V(6)
  FP = 1. + DPP
  BR = FP * BR0
  RHO = FP * R0

```

```

  ARG0 = PI02 + PHI0
  R(1) = X0 * CDEXF(XI*ARG0)

```

```

ARG1 = PHI0 + AL0
D1 = X0 * SNPHI0 / DCOS(ARG1)
R(2) = D1 * CDEXP(XI*ARG1)

REAL8 = DSIN(ARG1)
GAM0 = DARSIN( (L/RHO) - REAL8)
THETA = ARG1 + GAM0
ARG2 = ARG1 - PIO2
R(3) = RHO * CDEXP(XI*ARG2)

```

```

ARG3 = PIO2 - GAM0
R(4) = RHO * CDEXP(XI*ARG3)

```

```

Z = R(1) + R(2) + R(3) + R(4)
H = DIMAG(Z)
COSPHG=DCOS(PHI0-GAM0)
D4=H*SNPHI0/COSPHG
X0F=H*DCOS(GAM0)/COSPHG
AL0F = AL0 + 2.*PHI0 - THETA
S = RHO * THETA

```

```

TAR(1) = S
TAR(2) = 0.
TAR(3) = BR
TAR(4) = B0
TAR(5) = ARG1 * DEGRAD
TAR(6) = GAM0 * DEGRAD

```

```

DO 1 I=1,3
1  ML(I) = MEND - 2 - I
   CALL DRIFT(ML(1),D1)
   CALL MAGNET(ML(2),TAR)
   CALL DRIFT(ML(3),D4)
   CALL MMM(M,3,ML)

   CALL RTRV7(M,T,RW)
   CALL MXV7(T,V,V)
   V(1) = X0F
   V(2) = AL0F
   V(5) = V(5) - D1 - S - D4 + S0
   IF (B0.GE.0.) GO TO 4
   V(1) = -X0F
   V(2) = -AL0F
4  RETURN

```

```

2  CALL MAGNET(M,PAR)
   RETURN
   END

```

```

C ***** BASIS *****
C SUBROUTINE BASIS(X,FI,NN)
C IMPLICIT REAL*8 (A-H,O-Z)
C >>> FOR EVALUATION OF BESSEL FUNCTION "FI" ONLY
C >>> DIMENSION OF FI IN CALLING PROGRAM MUST BE GE 2
C

```

```

    DIMENSION FI(1), PI(600)
    NMAX = JMAX0((NN+1),2)
    KMAX=NMAX
    SUM=0.
    I=X
    JMAX=I+31
    TZ=2.D0/X
    JM2=JMAX+2
    DO 10 J=JM2,NMAX
10  PI(J)=0.
    JJMAX = JMAX + 1
    PI(JJMAX)=1.D-38
    DO 11 J=1,JMAX
    K=JMAX+2-J
    DK=K-1
    PI(K-1)=DK*TZ*PI(K)+PI(K+1)
11  SUM=SUM+PI(K)
    SUM=SUM+SUM
    A=DEXP(X)/(PI(1)+SUM)
    DO 12 N=1,NMAX
12  FI(N)=A*PI(N)
    RETURN
    END

C ***** BET *****
    SUBROUTINE BET(M,BETX,BETY,IF)
C  COMPUTES BETATRON FUNCTIONS
C
C      BETX(I)=MU,BETA,ALPHA,GAMMA,XEQ,DXEQ   FOR RX
C      BETY(I)=MU,BETA,ALPHA,GAMMA,YEQ,DYEQ   FOR RY
    IMPLICIT REAL*8 (A-H,O-Z)
    INCLUDE 'SWTCH.CCC'
    INCLUDE 'CONST.CCC'

    DIMENSION RX(2,3),RY(2,3),BETX(9),BETY(9),RXI(2,3),RYI(2,3)
    DIMENSION RW(3),RWI(3)
C
    QINV = 0.
    NN = -M
    CALL RXY(M,RX,RY,RW)
    TR= RX(1,1) + RX(2,2)
    CS=TR/2.D0
    BETX(9) = CS
    IF(TR.EQ.2.D0)GO TO 1
    IF(TR.EQ.(-2.D0))GO TO 2
    IF (DABS(TR).GT.2.D0)GO TO 3
    GO TO 7
3  IF( IF.EQ.0 ) GO TO 33
    BETX(1)=0.
    GO TO 4
1  BETX(1)=0.
    GO TO 4
2  BETX(1)=PI
4  DO 5 I=2,6
5  BETX(I)=0.
    GO TO 6
33 SN =DSIGN(1.D0, TR )*DSQRT( CS**2 - 1.D0 )

```

```

      S = DLOG(DABS( CS ) + SN )
      BS = RX(1,2)
      GO TO 8
7    BS = RX(1,2)
      SN=DSQRT(1.D0-CS*CS)*BS/DABS(BS)
      S=DATAN2(SN,CS) + TPI
8    B=BS/SN
      IF (DABS(TR) .GT. 2.D0) B=-DABS(B)
      G=-RX(2,1)/SN
      A=(RX(1,1)-RX(2,2))/(2.D0*SN)
      BETX(1)=S
      BETX(2)=B
      BETX(3)=A
      BETX(4)=G
      CALL RXY(NN,RXI,RYI,RWI)
      QINV=1.D0
      CSS = 2.D0*( 1.D0 - CS )
      BETX(5) = (RX(1,3)+RXI(1,3)) / CSS
      BETX(6) = (RX(2,3)+RXI(2,3)) / CSS
6    TR = (RY(1,1) + RY(2,2))
      CS=TR/2.D0
      BETY(9) = CS
      IF(TR.EQ.2.D0)GO TO 11
      IF(TR.EQ.(-2.D0))GO TO 12
      IF (DABS(TR).GT.2.D0)GO TO 13
      GO TO 17
13  IF( IF.EQ.0 ) GO TO 313
      BETY(1)=0.
C   INDEX M HAS ITS OWN INFO, WHICH REFERENCES DATA AND MATRIX LOC.OF N.
      GO TO 14
11  BETY(1)=0.
      GO TO 14
12  BETY(1)=PI
14  DO 15 I=2,6
15  BETY(I)=0.
      GO TO 16
313 SN =DSIGN(1.D0, TR )*DSQRT( CS**2 - 1.D0 )
      S = DLOG(DABS( CS ) + SN )
      BS = RY(1,2)
      GO TO 20
17  BS = RY(1,2)
      SN=DSQRT(1.D0-CS*CS)*BS/DABS(BS)
      S=DATAN2(SN,CS)+ TPI
20  B=BS/SN
      IF (DABS(TR) .GT. 2.D0) B=-DABS(B)
      G = -RY(2,1)/SN
      A = (RY(1,1)-RY(2,2))/(2.D0*SN)
      BETY(1)=S
      BETY(2)=B
      BETY(3)=A
      BETY(4)=G
      IF(QINV)18,18,19
18  CALL RXY(NN,RXI,RYI,RWI)
19  CSS = 2.D0*( 1.D0 - CS )
      BETY(5) = (RY(1,3)+RYI(1,3)) / CSS

```

```

      BETY(6) = (RY(2,3)+RYI(2,3)) / CSS
16  J=2
      BETX(7) =DSQRT(DABS( BETX(J) ) )
      BETY(7) =DSQRT(DABS( BETY(J) ) )
      BETX(8)=BETX(1)*180.D0/PI
      BETY(8)=BETY(1)*180.D0/PI
      RETURN
      END
C
C ***** BINSIZ *****
      SUBROUTINE BINSIZ(A1,A2,NAA,BL,BH,NB,BWID)
C      SUBROUTINE TO DETERMINE REASONABLE HISTOGRAM INTERVALS
C      GIVEN ABSOLUTE UPPER AND LOWER BOUNDS A1 AND A2
C      AND DESIRED MAXIMUM NUMBER OF BINS NAA
C      PROGRAM MAKES REASONABLE BINNING FROM BL TO BH OF WIDTH BWID
C      F. JAMES, AUGUST, 1974
C      TYPED BY E. CLOSE 9/75 FROM LISTING FROM CERN THAT CAME
C      WITH THE CODE.
      IMPLICIT REAL*8 (A-H,O-Z)
      AL= DMIN1(A1,A2)
      AH= DMAX1(A1,A2)
      IF(AL.EQ.AH) AH= AL+1.D0
C      IF(NAA.EQ.-1), PROGRAM USES BWID INPUT FROM CALLING ROUTINE
      IF(NAA.EQ.-1) GOTO 150
10    NA=NAA-1
      IF(NA.LT.1) NA= 1
C      GET NOMINAL BIN WIDTH IN EXPON FORM
20    AWID=(AH-AL)/DFLOAT(NA)
      LOG= DLOG10(AWID)
      IF(AWID.LE.1.0D0) LOG= LOG-1
      SIGFIG= AWID*(10.D0**(-LOG))
C      ROUND MANTISSA UP TO 2,2.5,5,OR 10
      IF(SIGFIG.GT. 2.0D0) GOTO 40
      SIGRND= 2.0D0
      GOTO 100
40    IF(SIGFIG.GT.2.5D0) GOTO 50
      SIGRND= 2.5D0
      GOTO 100
50    IF(SIGFIG.GT.5.0D0) GOTO 60
      SIGRND= 5.0D0
      GOTO 100
60    SIGRND=1.0D0
      LOG= LOG+1
100   CONTINUE
      BWID= SIGRND*10.D0**LOG
      GOTO 200
C      GET NEW BOUNDS FROM NEW WIDTH BWID
150   IF(BWID.LE.0.D0) GOTO 10
200   CONTINUE
      ALB= AL/BWID
      LWID= ALB
      IF(ALB.LT.0.0D0) LWID= LWID-1
      BL= BWID* DFLOAT(LWID)
      ALB= AH/BWID+1.0D0
      KWID= ALB

```



```

      IF(ALB.LT.0.0D0) KWID= KWID-1
      BH= BWID*DFLOAT(KWID)
      NB= KWID-LWID
      IF(NAA.GT.5) GOTO 240
      IF(NAA.EQ.-1) RETURN
C      REQUEST FOR ONE BIN IS DIFFICULT CASE
      IF(NAA.GT.1 .OR. NB .EQ. 1) RETURN
      BWID= BWID*2.0D0
      NB= 1
      RETURN
240  IF(2*NB.NE.NAA) RETURN
      NA= NA+1
      GOTO 20
      END

C ***** BMIS *****
      SUBROUTINE BMIS
      IMPLICIT REAL*8 (A-H,O-Z)
C  BEGIN MISALIGNMENT
      INCLUDE 'SWTCH.CCC'
      INCLUDE 'CONTRL.CCC'

      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      COMMON /VARIAT/ VT(15,15)
      INTEGER IBLNK/1H /,doIMIN/1H-/
C
      IF (MODE.EQ.3) GO TO 10
      ISGN=IBLNK
      IF (OP .LT. 0) ISGN=doIMIN
      WRITE (3, 20), ISGN
20  FORMAT (6H *** ,A1,8X,4HBMIS)
      IF (MODE.EQ.2) RETURN
10  BEND=0.
      RETURN
      END

C ***** BMLE *****
      SUBROUTINE BMLE(M)
C  REPLACE BEAM LINE
C  CREATES A NEW BEAM LINE IF BEAM LINE FUNCTION REFERENCE IS FBLE.
C  WHEN BEAM LINEA REFERENCE IS FBLE, THEN REPLACEMENTS ARE DONE AT
C  TIME THAT MILIST IS CREATED.
C .C  BML          QF  OO  QD  QD  OO  QF
C FX  FBLE  2      .C      QF  QD
C      BMLE  2      FX  .C1  QF1  QD1
C  CREATES
C .C1  BML          QF1  OO  QD1  QD1  OO  QF1

C
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      INCLUDE 'STORE.CCC'

      PARAMETER (NPMAX=300)
      DIMENSION NPAR(NPMAX),NP1(NPMAX/2),NP2(NPMAX/2),IC(24)
      INTEGER ELNUM,OLDBL,OPNUM,OPNAME,OP,BDAT
      EQUIVALENCE (OPNAME,IC(1)),(NAME,IC(2)),(OP,IC(3))

```

```

EQUIVALENCE (KA,IC(4)),(KB,IC(5))
DATA IBLNK/1H /

```

```

C GET M OF FUNCTION

```

```

MFN = MDAT(M,1)
KAF = INFF(4,MFN)
KA = INFF(4,M)
IF (KA.NE.KAF) GO TO 130

```

```

C RETRIEVE DATA. MAKE PARALLEL ARRAYS

```

```

C NP1 CONTAINS ELEMENTS TO BE REPLACED (FBLE DATA)
C NP2 CONTAINS REPLACEMENT ELEMENTS (BMLE DATA)

```

```

DO 10 I=1,KA
NP1(I) = BDAT(MFN,I+1)

```

```

10 NP2(I) = BDAT(M,I+2)

```

```

C RETRIEVE OLD BEAM LINE

```

```

OLDBL = BDAT(MFN,1)
NB1 = ELNUM(OLDBL)
NI = INFF(17,NB1)
IF (NI.GT.NPMAX) GO TO 135

```

```

CALL DATA (NB1,2,1,NI,NPAR)

```

```

C SEARCH NPAR FOR NAMES IN NP1. WHEN FOUND, REPLACE WITH NP2.

```

```

DO 20 I=1,NI
IF (NPAR(I).EQ.IBLNK) GO TO 20
DO 15 IB=1,KA
IF (NP2(IB).EQ.IBLNK) GO TO 15
IF (NPAR(I).EQ.NP1(IB)) NPAR(I) = NP2(IB)
15 CONTINUE
20 CONTINUE
DO 25 J=1,24
25 IC(I) = 0

```

```

C

```

```

C IF NEW BML NAME FIELD IS BLANK, USE ORIGINAL NAME AND INFF.

```

```

NEWBL = BDAT(M,2)
IF (NEWBL.EQ.' ') NEWBL = OLDBL
OPNAME = 'BML'
NAME = NEWBL
OP = INFF(3,M)
KA = INFF(4,M)
KB = INFF(5,M)
IF (NEWBL.NE.OLDBL) GO TO 125

```

```

C NEW BEAM LINE REPLACES OLD ONE.

```

```

CALL STDAT(NB1,2,1,NI,NPAR)
GO TO 126

```

```

C NEW BEAM LINE

```

```

125 MIN = MIN - 1
CALL INFW(MIN,8,M)
M1 = MIN
IC(21) = 'SNGL'
CALL WRTINF(IC,M1)

```

```

C MAKE INFF SAME AS IF DATA HAD BEEN READ WITH INDEFINITE LOAD

```

```

NI2 = NI*2
CALL RESRV6(M1,0,0,NI2,2,0)

```

```

      CALL STDAT(M1,2,1,NI,NPAR)
C  RESET INFF
      INFF(17,M1) = INFF(18,M1)
      INFF(19,M1) = 0
      INFF(24,M1) = 0
C  WRITE CARD IMAGE
126  NPR = 12
      IF (NPR.GT.NI) NPR = NI
      WRITE (3,106) NAME,OPNAME,KA,KB,(NPAR(I),I=1,NPR)
106  FORMAT (6H *** ,1X,A5,2X,A5,1X,I3,1X,I3,5H // ,12(A4,1X))
50   IF (NPR.GE.NI) GO TO 110
      II = NPR + 1
      NPR = NPR + 12
      IF (NPR.GT.NI) NPR=NI
      WRITE (3,107) (NPAR(I),I=II,NPR)
107  FORMAT (6H * ,21X,5H // ,12(A4,1X))
      GO TO 50
C  READJUST M
110  M = M1
      GO TO 120
130  NAMF = BDAT(M,1)
      NAM = INFF(2,M)
      WRITE (3,132) NAMF,NAM
132  FORMAT (5X,'*** ERROR *** NUMBER OF REPLACEMENTS IN FBLE ',A5,
1    ' ' AND BMLE',A5,' SHOULD BE EQUAL.')
      CALL EXIT
135  WRITE (3,133)
133  FORMAT (5X,'*** ERROR *** NUMBER OF ELEMENTS IN BML TOO LARGE
1    FOR DIMENSION IN BMLE SUBROUTINE.'/ 5X,'ENLARGE PARAMETER
2    NPMAX IN SUBROUTINE BMLE.')
      CALL EXIT

120  RETURN
      END

```

```

C ***** BVAL *****
      SUBROUTINE BVAL(M)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C  PARTICLE BEAM DEFINITION
C  COMPUTES KINEMATICS AND UNNORMALIZED EMITTANCES.
C  IF INPUT EMITTANCES ARE NORMALIZED, UNNORMALIZED VALUES
C  WILL BE CALCULATED.
C  UNNORMALIZED VALUES WILL BE STORED FOR LATER USE.

```

```

C.....
C  NAM  BVAL  KA  KB// PT  A  EPSX  EPSY  EPSL  SIGL
C.....
C  NAM  BVAL      //      EPSX  EPSY  EPSL  SIGL
C  KIN  BVAL      P  A  EPSX  EPSY  EPSL  SIGL
C  KNM  BVAL  1  1// T  A  EPSX  EPSY  EPSL  SIGL

```

```

C
C      PT = EITHER P, T OR BLANK WHERE
C      P = MOMENTUM GEV/C
C      T = KINETIC ENERGY GEV

```

```

C      A = ATOMIC WEIGHT
C      IF A = 0 OR BLANK, ELECTRON MASS IS ASSUMED.
C      EMITTANCES (EPS) IN MM-MRAD
C      SIGL (MM)
C
C      IF KA=0 OR BLANK, 1ST PARAMETER IS P.
C      IF KA NOT 0 (OR BLANK), 1ST PARAMETER IS T.
C
C      IF KB = 0 (OR BLANK), THE NEXT CALL TO CYC WILL
C      PRINT ETAS = -VP(5)
C      IF KB NOT 0 , THE NEXT CALL TO CYC WILL
C      PRINT ETAT =VP(5) -S/GAMMASQ.
C
C      UNNORMALIZED INPUT — WHEN PT=0, THE EPSX ETC. ON THE INPUT CARD
C      REPRESENT UNNORMALIZED EMITTANCES AND NO
C      KINEMATIC CALCULATIONS ARE DONE.
C      NORMALIZED INPUT — WHEN PT NOT 0 AND EPSX,ETC. ARE PRESENT.
C
C      INCLUDE 'BINFF.CCC'
C      INCLUDE 'KINET.CCC'
C
C      DIMENSION PAR(6),EM(4)
C      EQUIVALENCE (EPSX,EM(1)),(EPSY,EM(2)),(EPSL,EM(3)),(SIGP,EM(4))
C      DATA E0,RME,CONK/.9382796D0,5.110034D-4,3.33564095D0/
C
C      KA = INFF(4,M)
C      KB = INFF(5,M)
C      CALL DATA (M,1,1,6,PAR)
C      IF (PAR(1).EQ.0.) GO TO 22
C      A = PAR(2)
C      IF (A.EQ.0.) A = RME
C      AMPC2 = A*E0
C      AMPC2R = 1.D0/AMPC2
C      IF (KA.NE.0) GO TO 15
C 1ST PARAM IS P
C      P = PAR(1)
C      BETGAM = P*AMPC2R
C      GAM = DSQRT(BETGAM*BETGAM + 1.D0)
C      TK = AMPC2*(GAM-1.D0)
C      GO TO 20
C 1ST PARAM IS T
15  TK = PAR(1)
C      GAM = TK*AMPC2R + 1.D0
C      BETGAM = DSQRT(GAM*GAM - 1.D0)
C      P = AMPC2*BETGAM
20  BETT = BETGAM/GAM
C      BRHO = P*CONK
C      IF (KB.NE.0) KNFLAG = .TRUE.
C      WRITE (3,101)
C      CALL DASH
C      WRITE (3,100) P,TK,A,BRHO,BETT,GAM,BETGAM
C      IF ((PAR(3).EQ.0.).AND.(PAR(4).EQ.0.).AND.(PAR(5).EQ.0.).AND.
1  (PAR(6).EQ.0.)) RETURN

```

```

BETGAMR = 1.D0/BETGAM
EPSX = PAR(3)*BETGAMR
EPSY = PAR(4)*BETGAMR
EPSL = PAR(5)*BETGAMR
GO TO 25

```

C ALREADY UNNORMALIZED CASE

```

22 EPSX = PAR(3)
   EPSY = PAR(4)
   EPSL = PAR(5)
25 SIGL = PAR(6)
   SIGP = EPSL/SIGL
   CALL STDAT (M,5,1,4,EM)
   WRITE (3,105) EPSX,EPSY,EPSL,SIGL,SIGP
   CALL DASH

```

C

```

100 FORMAT (8X,'P = ',F10.5,' GEV/C',6X,'T = ',
1   F10.5,' GEV',12X,'A = ',F10.5,5X,'BRHO = ',F10.5,/
2   5X,'BETA = ',E15.8,3X,'GAMMA = ',E15.8,3X,
3   'BETAGAMMA = ',E15.8)
101 FORMAT (1H )
105 FORMAT (/5X,'EMITTANCES UNNORMALIZED (MM-MRAD)'/
1   8X,'EPSX = ',F10.6,4X,'EPSY = ',F10.6,4X,'EPSL = ',
2   F10.6,10X,'SIGL = ',F10.6,1X,'MM',4X,'SIGP = ',F10.6,
3   ' (0/00)')
   RETURN
   END

```

C.....CALC.....

```

SUBROUTINE CALC(M)
IMPLICIT REAL*8(A-H,O-Z)

```

C INSTRUCTION CALC — SIMULATES AN H-P CALCULATOR.

C COMMANDS ARE ENTERED AS 5 CHARACTER BCD DATA. COMMANDS ALLOWED —

C	1/X	SQRT	LN	Y**X X**X	EXP
C	X-Y	RDN	SIN COS TAN	RUP	ASIN ACOS ATAN
C		STO	RCL	ABS	
C	ENTP	CHS	EEX	CLX	
C	-			X=Y	
C	+	P-R	R-P	XLEY	
C	*			XGTY	
C	/	PI	LSTX	X=0	

C KA CONTROLS PRINTING AS FOLLOWS — KA = 0 — NO PRINT

C KA = 1, PRINT STACK AT END

C KA = 2, PRINT STACK EVERY STEP

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
INCLUDE 'CONTRL.CCC'
INCLUDE 'MATCH3.CCC'

```

```

COMMON/CALC/REG(109)
DIMENSION CMND(16,5),LX(16,5),NUM(14),STK(5)
LOGICAL LIFT,SKIP,NPRT

```

```

INTEGER CMND,ALPH,BDAT
COMPLEX CX
EQUIVALENCE (STK,REG),(X,STK(1)),(Y,STK(2)),(Z,STK(3)),(T,STK(4)),
              (XL,STK(5)),(CX,X)
DATA NUM/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H ,1H.,1H+,1H-/
DATA (LX(1,L),L=1,5)/0,1,1,0,1/
DATA (LX(2,L),L=1,5)/0,0,1,1,1/
DATA (LX(3,L),L=1,5)/0,0,0,0,1/
DATA (LX(4,L),L=1,5)/5*0/
DATA (LX(5,L),L=1,5)/1,4*0/
DATA (LX(6,L),L=1,5)/1,0,1,1,0/
DATA (LX(7,L),L=1,5)/1,4*0/
DATA (LX(8,L),L=1,5)/1,4*0/
DATA (LX(9,L),L=1,5)/0,1,1,0,1/
DATA (LX(10,L),L=1,5)/0,0,1,1,1/
DATA ((LX(K,L),K=11,16),L=1,5)/30*0/
DATA (CMND(1,L),L=1,5)/4H ,4H1/X ,4HSQRT,4H ,4HLN /,
2    (CMND(2,L),L=1,5)/4HX-Y ,4HRDN ,4HSIN ,4HCOS ,4HTAN /,
3    (CMND(3,L),L=1,5)/4H ,4H ,4HSTO ,4HRCL ,4HABS /,
4    (CMND(4,L),L=1,4)/4HENTR,4HCHS ,4HEEX ,4HCLX /,
5    CMND(5,1) /4H- /,
6    (CMND(6,L),L=1,4)/4H+ ,4H ,4HP-R ,4HR-P /,
7    CMND(7,1) /4H* /,
8    (CMND(8,L),L=1,3)/4H/ ,4HPI ,4HLSTX /,
9    (CMND(9,L),L=1,5)/4H ,4HY**X,4HX*X ,4H ,4HEXP /,
A    (CMND(10,L),L=1,5)/4H ,4HRUP ,4HASIN,4HACOS,4HATAN/,
C    CMND(12,4) / 4HCLST /,
D    (CMND(K,1),K=13,16)/4HX=Y ,4HXLEY,4HXGTY,4HX=0 /
DATA IBLK/4H /

SKIP = .FALSE.
LIFT = .TRUE.
NPRT = .FALSE.
NBC = INFF(17,M)
KA = INFF(4,M)
ZR = 0.
P1 = DACOS(ZR)
PI = 2.D0*P1
I = 0
ICMD = 5H
IF (NOPR.OR.KA.EQ.0) NPRT = .TRUE.
IF (.NOT.NPRT) CALL DASH
IF (.NOT.NPRT) WRITE(3,901)
IF (.NOT.NPRT) CALL DASH
IF (.NOT.NPRT.AND.KA.EQ.2) WRITE(3,902) I,ICMD,X,Y,Z,T,XL

DO 300 I=1,NBC
  ICMD = BDAT(M,I)
  IF (ICMD.EQ.IBLK) GO TO 300
  IF (.NOT.SKIP) GO TO 2
  SKIP = .FALSE.
  GO TO 3

```

```

      DO 6 L=1,5
      KL = 5*(K-1) + L
      IF(ICMD.EQ.CMND(K,L)) GO TO 77
6     CONTINUE

      DECODE (4,501,ICMD) IC1
501  FORMAT (A1,3X)

      DO 8 K=1,14
      IF(IC1.EQ.NUM(K)) GO TO 500
8     CONTINUE

9     WRITE (3,10) ICMD
10    FORMAT(1X,5H**** ,A5,'IS NOT A RECOGNIZED CALC COMMAND')
      ERROR = .TRUE.
      RETURN

C  FLOATING POINT NUMBER
500  DECODE(4,900,ICMD) F
900  FORMAT(F5.0)
      IF (LIFT) CALL STKLFT(X,Y,Z,T)
      X = F
      GO TO 1
77   IF(LX(K,L).EQ.1) XL=X
7    GO TO ( 1,12,13, 1,15, 21,22,23,24,25, 1, 1,33,34,35,
.           41,42,43,44, 1, 51, 1, 1, 1, 1, 61, 1,63,64, 1,
.           71, 1, 1, 1, 1, 81,82,83, 1, 1, 1,92,93, 1,95,
.           1,102,103,104,105, 1, 1, 1, 1, 1, 1, 1, 1,124, 1,
.           131, 1, 1, 1, 1, 141, 1, 1, 1, 1, 151, 1, 1, 1, 1,
.           161, 1, 1, 1, 1 ) ,KL

* 1/X
12   X=1.D0/X
      GO TO 1

* SQRT
13   X = DSQRT(X)
      GO TO 1

* LN
15   X = DLOG(X)
      GO TO 1

* X-Y
21   XX=X
      X = Y
      Y = XX
      GO TO 1

* RDN
22   XX=X
      X=Y
      Y=Z
      Z=T
      T=XX
      GO TO 1

* SIN
23   X = DSIN(X)
      GO TO 1

```

```

* COS
24  X = DCOS(X)
    GO TO 1

* TAN
25  X = DTAN(X)
    GO TO 1

* STO
33  CALL STOREC(M,I,NREG,NSY)
    IF (NREG.LT.0) GO TO 331
    REG(NREG) = X
    GO TO 5
331 STORE(NSY) = X
    GO TO 5

* RCL
34  CALL STOREC(M,I,NREG,NSY)
    IF (LIFT) CALL STKLFT(X,Y,Z,T)
    IF (NREG.LT.0) GO TO 341
    X = REG(NREG)
    GO TO 5
341 X = STORE(NSY)
5   SKIP = .TRUE.
    GO TO 1

* ABS
35  X = DABS(X)
    GO TO 1

* ENT
41  CALL STKLFT(X,Y,Z,T)
    LIFT = .FALSE.
    GO TO 3

* CHS
42  X = -X
    GO TO 1

* EEX
43  ICMD2 = BDAT(M,I+1)
    DECODE (4,900,ICMD2) XEXP
    X = X*10.D0**XEXP
    SKIP = .TRUE.
    GO TO 1

* CLX
44  X = 0.
    LIFT = .FALSE.
    GO TO 3

* -
51  X = Y - X
    GO TO 4

* +
61  X = Y + X
    GO TO 4

* P-R
63  X1 = X*DCOS(Y)
    Y = X*DSIN(Y)
    X = X1
    GO TO 1

* R-P
64  TH = DATAN2(Y,X)

```



```

      R = CABS(CX)
      X = R
      Y = TH
      GO TO 1
• •
71  X = Y * X
    GO TO 4
• /
81  X = Y / X
    GO TO 4
• PI
82  IF (LIFT) CALL STKLFT(X,Y,Z,T)
    X = PI
    GO TO 1
• LASTX
83  IF(LIFT) CALL STKLFT(X,Y,Z,T)
    X = XL
    GO TO 1
• Y**X
92  X = Y**X
    GO TO 4
• X*X
93  X = X*X
    GO TO 1
• EXP
95  X = DEXP(X)
    GO TO 1
• RUP
102 XX=X
    X = T
    T = Z
    Z = Y
    Y = XX
    GO TO 1
• ASIN
103 X = DASIN(X)
    GO TO 1
• ACOS
104 X = DACOS(X)
    GO TO 1
• ATAN
105 X = DATAN(X)
    GO TO 1
• CLST
124 X = 0.
    Y = 0.
    Z = 0.
    T = 0.
    GO TO 1
• X=Y
131 IF(X.NE.Y) SKIP = .TRUE.
    GO TO 1
• XLEY
141 IF(X.GT.Y) SKIP = .TRUE.
    GO TO 1

```

```

* XGTY
151 IF(X.LE.Y) SKIP = .TRUE.
    GO TO 1
* X=0
161 IF(X.NE.0.)SKIP = .TRUE.
    GO TO 1

C DROP STACK
4   Y = Z
    Z = T

1   LIFT = .TRUE.
3   IF (.NOT.NPRT.AND.KA.EQ.2) WRITE(3,902) I,ICMD,X,Y,Z,T,XL
300 CONTINUE
    CALL STDAT (M,5,1,5,STK)

    IF (.NOT.NPRT.AND.KA.EQ.1) WRITE(3,902) I,ICMD,X,Y,Z,T,XL
    IF (.NOT.NPRT) CALL DASH

901 FORMAT (4X,1HI,5X,5HCMND ,9X,1HX,14X,1HY,14X,1HZ,14X,1HT,14X,1HL)
902 FORMAT (1X,I4,5X,A5,5F15.5)
    RETURN
    END

C ***** CALFCN *****
    FUNCTION CALFCN(PVEC)
    IMPLICIT REAL*8 (A-H,O-Z)
CC      CALLED ONLY FROM IMPROV. TRANSFORMS THE FUNCTION FCN
CC      BY DIVIDING OUT THE QUADRATIC PART IN ORDER TO FIND FURTHER
CC      MINIMA. CALCULATES (F-FMIN)/(X-XMIN)*V*(X-XMIN)
CC
    DIMENSION PVEC(15)
    COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
    INCLUDE 'PAREXT.CCC'
    COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
    COMMON /DERIVA/ G(30) ,G2(30)
    COMMON /VARIAT/ VT(15,15)
    CALL INTOEX(PVEC)
    CALL FCN(NPAR,G,F,U,4)
    NFCN = NFCN + 1
    DO 200 I= 1, NPAR
    G(I) = 0.D0
    DO 200 J= 1, NPAR
200 G(I) = G(I) + VT(I,J) * (XT(J)-PVEC(J))
    DENOM = 0.D0
    DO 210 I= 1, NPAR
210 DENOM = DENOM + G(I) * (XT(I)-PVEC(I))
    IF (DENOM .LE. 0.D0) ISW(2) = 0
    IF (DENOM .LE. 0.D0) DENOM = 1.D0
    CALFCN = (F-APSI) / DENOM
    RETURN
    END

C ***** CELL *****
    SUBROUTINE CELL(M,PAR)
    IMPLICIT REAL*8 (A-H,O-Z)
C MAGNET PARAMETER GAMMA IS DEPENDENT ON E. GAMMA IS P(5)

```

```

C SET UP MATRICES FOR AN FDDF CELL (WITH DRIFT SPACES).
C PARAMETER LIST...LMAG,K,RHO,LSTR,LSEP,E,OMEGA
C CYCLE MATRIX PRODUCT (N+1-1)=(I-1)(I-2)...(K)(K-1)...(I),I=1,K
C COMPUTE THE PRODUCT (FIRST CYCLED MATRIX IF L.NE.0)
C IF WS=0,STORE THE CYCLED MATRICES IMMEDIATELY AFTER MATRIX M.
C IF WS.NE.0,STORE THE CYCLED MATRICES IN WORK SPACE.
C QF = FOCUSING MAGNET = MATRIX M-13
C QD = DEFOCUSING MAGNET = MATRIX M-14
C QS = HALF THE CENTER DRIFT SPACE = MATRIX M-15
C QT = HALF THE DRIFT SPACE BETWEEN A D AND AN F MAGNET = MATRIX M-16
C SUBROUTINE GENERATES 17 DRIFT MATRICES AS FOLLOWS...
C THE FDDF CELL = Q = THE PRODUCT OF...QS QF QT QT QD QS QS QD QT QT QF
C QS = MATRIX M
C QI = THE I-TH CYCLED TRANSFER MATRIX = MATRIX M-1, WHERE
C I = 1,12, WHERE Q1 = Q.
C MAGNET PARAMETER DELTA IS SET = 0.
      INCLUDE 'BMI.CCC'
      INCLUDE 'SWTCH.CCC'

      DIMENSION PAR(7),P(7)
      P(1)=PAR(1)
      P(3)=PAR(3)
      SOV2=PAR(4)/2.
      TOV2=PAR(5)/2.
      P(2)=DABS(PAR(2))
      P(4)=PAR(7)
      P(5)=PAR(6)

C SET UP ENTRANCE AND EXIT ANGLES EQUAL TO EACH OTHER.
C SAME CONVENTION HOLDS AS FOR MAGNET.
      P(6)=P(5)

C COMPUTE MAGNETS AND SET UP MI LIST, STORE MAGNET PARAMETERS.
C SET UP QF
      L=M-13
      CALL STDAT(L,1,1,6,P)
      CALL MAGNET(L,P)
      MI(2)=L
      MI(11)=L

C SET UP QD
      L=M-14
      P(2)=-P(2)
      CALL STDAT(L,1,1,6,P)
      CALL MAGNET(L,P)
      MI(5)=L
      MI( 8)=L

C SET UP QS
      L=M-15
      CALL DRIFT(L,SOV2)
      MI(1)=L
      MI( 6)=L
      MI( 7)=L
      MI(12)=L

C SET UP QT
      L=M-16
      CALL DRIFT(L,TOV2)
      MI(3)=L

```

```

MI(4)=L
MI( 9)=L
MI(10)=L
L=M-1
CALL MM(M,12,MI)
C IF CYCSWT IS ON, COMPUTE CELL MATRIX ONLY FOR TEMPORARY USE
C                                     (FOR DESIGN INSTRUCTIONS).
      IF(CYCSWT) RETURN
C COMPUTE 12 CYCLED MATRICES
      CALL CYCLE (M,12,MI,0,0)
C STORE THE MILIST AS DATA FOR Q
      CALL STDAT(L,3,1,12,MI)
      RETURN
      END
C ..... CHKD .....
      SUBROUTINE CHKD(NFT,KN,ITP)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'FLTN.CCC'
      INTEGER ELNUM
      DATA MASKF/ZFF000000/
      INTEGER doISCAL/4HSCAL/,doISNGL/4HSNGL/

C DETERMINE KIND AND TYPE OF NFT
      MSK = LAND(MASKF,NFT)
      IF (MSK.NE.IFL(1).AND.MASK.NE.IFL(12)) GO TO 12
      CALL SHIFT (NFT,-8,MSK)
      MSK = LAND(MASKF,MSK)
12   DO 15 L=2,14
      IF (MSK.EQ.IFL(L)) GO TO 25
15   CONTINUE
C SYMBOLIC. DETERMINE KIND
      IK = ELNUM(NFT)
      KN = INFF(20,IK)
      ITP = INFF(21,IK)
      GO TO 30
C FLOATING POINT. SCALAR AND SINGLE
25   KN = doISCAL
      ITP = doISNGL
30   CONTINUE
      RETURN
      END
C ..... CHRM .....
      SUBROUTINE CHRM(GK,RI,SS,TAN1, TAN2,BETX,BETY,BXP,BYP,CRX,CRY)
      IMPLICIT REAL*8 (A-H,O-Z)
C
      DIMENSION BETX(6),BETY(6),BXP(6),BYP(6),
1 BX1(6),BY1(6),BX2(6),BY2(6)
      CRX=0.
      CRY=0.
      IF(GK.EQ.0..AND.RI.EQ.0.) RETURN
      IF(RI.NE.0.) GO TO 11
      CRX = (BETX(4)+GK*BETX(2))*SS + BETX(3) - BXP(3)
      CRY = (BETY(4)-GK*BETY(2))*SS + BETY(3) - BYP(3)

```

```

      RETURN
11  CONTINUE
      DO 1 I=1,6
      BX1(I)=BXP(I)
      BY1(I)=BYP(I)
      BX2(I)=BETX(I)
1    BY2(I)=BETY(I)
      BX1(3)=BX1(3)-TAN1*RI*BX1(2)
      BX2(3)=BX2(3)+TAN2*RI*BX2(2)
      BY1(3)=BY1(3)+TAN1*RI*BY1(2)
      BY2(3)=BY2(3)-TAN2*RI*BY2(2)
      BX1(6)=BX1(6)+TAN1*RI*BX1(5)
      BX2(6)=BX2(6)-TAN2*RI*BX2(5)
      RI2=RI**2
      GKX=GK+RI2
      BKGX = GKX*BX1(2) + (1.D0+BX1(3)**2)/BX1(2)
      BKGX = -GK*BY1(2) + (1.D0+BY1(3)**2)/BY1(2)
      F2 = (SS*BKGX+BY2(3)-BY1(3))/2.D0
      IF (GKX.NE.0.) GO TO 12
      F5 = SS*( BX1(5)+BX2(5)-SS**2*RI/6.D0 )/2.D0
      F1 = SS*( BX1(2)+BX2(2)+SS*(BX2(3)-BX1(3))/3.D0 )/2.D0
      F3 = ( (BX1(2)+BX2(2))*(BX1(5)+BX2(5))
1        + BX1(2)*BX2(5) + BX2(2)*BX1(5) )/3.D0
2        + (BX2(3)-BX1(3))*(5.D0*(BX1(5)+BX2(5))-RI*SS**2)*SS/30.D0
3        - (BX1(2)+BX2(2))*RI*SS**2/12.D0
      F3 = F3*SS/2.D0
      GO TO 13
12   F1 = (SS*BKGX+BX2(3)-BX1(3))/GKX/2.D0
      F5 = (RI*SS-BX2(6)+BX1(6))/GKX
      F3= (2.D0*BKGX*F5-RI*F1 + BX2(6)*BX2(2) + 2.D0*BX2(3)*BX2(5)
1        - BX1(6)*BX1(2) - 2.D0*BX1(3)*BX1(5) )/3.D0/GKX
13   CRX = - (GKX+RI2)*F1 + RI*(GK+GK)*F3 + RI*BKGX*F5
1     - RI*(BXP(2)*BXP(6)-BETX(2)*BETX(6))
      CRY = - F2 + RI*BKGX*F5 + RI*(BYP(2)*BXP(6)-BETY(2)*BETX(6))
      IF (TAN1.EQ.0..AND.TAN2.EQ.0.) GO TO 14
      T1 = RI*TAN1
      T2 = RI*TAN2
      CRX = CRX + BXP(2)* ( T1-TAN1*((GK+GKX)*BXP (5)-T1*BXP (6)) )
1      + BETX(2)* ( T2-TAN2*((GK+GKX)*BETX(5)+T2*BETX(6)) )
2      + (BXP(3)+BX1(3))*BXP(5)*T1*TAN1 -(BETX(3)+BX2(3))*BETX(5)*T2*TAN2
      CRY = CRY + BYP (2)*(-T1+TAN1*(2.*GK*BXP (5)+T1*BXP (6)) )
1      + BETY(2)*(-T2+TAN2*(2.*GK*BETX(5)-T2*BETX(6)) )
2      -(BYP(3)+BY1(3))*BXP(5)*T1*TAN1+(BETY(3)+BY2(3))*BETX(5)*T2*TAN2
14   CONTINUE
      CRX=-CRX - CRX
      CRY = -CRY - CRY
      RETURN
      END
C ***** CMXDIV *****
      SUBROUTINE CMXDIV(V,IN,IM,DET,IDIM)
      IMPLICIT REAL*8 (A-H,O-Z)
C    REPLACES B BY (A INVERSE)*B. DESTROYS A.
C    A IS N BY N, B IS N BY M. A AND B ARE STORED IN V(N,N+M).
      COMPLEX*16 SAVE,PIVOT,DETERM,DET,V,CABS
      DIMENSION V(IDIM,1)

```

```

      INTEGER P
      N=IN
      M=IM
C
      NM1=N-1
      NP1=N+1
      NPM=N+M
      DETERM=(1.000,0.000)
      P=1
202 CONTINUE
      IMIN=P+1
C
      FIND PIVOT
      RPIVOT=0.0E0
      DO 209 I=P,N
      RSAVE=CDABS(V(I,P))
      IF (RSAVE .LE. RPIVOT) GO TO 209
      RPIVOT=RSAVE
      IBIG=I
209 CONTINUE
      IF (RPIVOT .GT. 0.0E0) GO TO 210
      DETERM=(0.0E0,0.0E0)
      GO TO 290
C
      ROW EXCHANGE AND ROW REDUCTION
210 PIVOT=V(IBIG,P)
      DETERM=DETERM*PIVOT
      DO 219 J=P,NPM
      SAVE=V(IBIG,J)/PIVOT
      V(IBIG,J)=V(P,J)
      V(P,J)=SAVE
219 CONTINUE
      IF (IBIG .NE. P) DETERM=-DETERM
      IF (P .GE. N) GO TO 250
      V(P,P) =(1.0E0,0.0E0)
C
      BLOCK REDUCTION
      DO 239 I=IMIN,N
      DO 238 J=IMIN,NPM
      V(I,J)=V(I,J)-V(I,P)*V(P,J)
238 CONTINUE
      V(I,P)=(0.0E0,0.0E0)
239 CONTINUE
      P=P+1
249 GO TO 202
C
      BACK SUBSTITUTION
250 CONTINUE
      IF (M .LE. 0) GO TO 290
      DO 259 K=NP1,NPM
      DO 253 P=1,NM1
      I=N-P
      DO 252 J=I,NM1
      V(I,K)=V(I,K)-V(J+1,K)*V(I,J+1)
252 CONTINUE
253 CONTINUE
259 CONTINUE
C
290 CONTINUE

```

```

      DET=DETERM
      RETURN
      END
C ***** CNVMAT *****
      SUBROUTINE CNVMAT(KN,T,RX,RY,RW,RPQ)
      IMPLICIT REAL*8 (A-H,O-Z)

C   IF KN=3, CONVERTS 7X7 MATRIX TO A 3X3
C   IF KN=7, CONVERTS 3X3 MATRIX TO A 7X7
      INCLUDE 'CONTRL.CCC'

      DIMENSION T(1),T1(49),T2(7,7),RX(2,3),RY(2,3),RW(3),RPQ(4)
      EQUIVALENCE(T1(1),T2(1))

      DO 1 I=1,49
        T1(I) = T(I)
1     CONTINUE
      IF (KN.EQ.7) GO TO 15
      IF (KN.NE.3) GO TO 30
C     CONVERT T ARRAY OF 7X7 TO RX,RY OF 3X3
      DO 3 J=1,3
        DO 2 I=1,2
          RX(I,J) = 0.
          RY(I,J) = 0.
2     CONTINUE
3     CONTINUE
      DO 8 I=1,2
        DO 7 J=1,2
          RX(I,J) = T2(I,J)
          RY(I,J) = T2(I+2,J+2)
7     CONTINUE
8     CONTINUE
      DO 9 I=1,2
        RX(I,3) = T2(I,6)
        RY(I,3) = T2(I+2,6)
9     CONTINUE
      RW(3) = T2(5,6)
C     EXTRACT P1,P2,Q1,Q2
      DO 10 I=1,4
10    RPQ(I) = T2(5,I)
      RETURN
C     CONVERT RX,RY OF 3X3 TO T OF 7X7
15    CONTINUE
      DO 17 I=1,49
        T(I) = 0.
        T1(I) = 0.
17    CONTINUE
      DO 20 I=1,2
        DO 19 J=1,2
          T2(I,J) = RX(I,J)
          T2(I+2,J+2) = RY(I,J)
19    CONTINUE
20    CONTINUE
      DO 21 I=1,2
        T2(I,6) = RX(I,3)

```

```

      T2(I+2,6) = RY(I,3)
21  CONTINUE
      T2(5,6) = RW(3)
      T2(5,5) = 1.
      T2(6,6) = 1.
      T2(7,7) = 1.
*    SET P1,P2,Q1,Q2
      DO 22 I=1,2
        T2(5,I) = T2(1,6)*T2(2,I) - T2(2,6)*T2(1,I)
        T2(5,I+2) = T2(3,6)*T2(4,I+2) - T2(4,6)*T2(3,I+2)
        RPQ(I) = T2(5,I)
22  RPQ(I+2) = T2(5,I+2)
      DO 26 I=1,49
        T(I) = T1(I)
26  CONTINUE
      RETURN

C          ERROR IF KN IS NOT MXY OR R7
30  WRITE (3,100) KN
100  FORMAT (5H *** , 'ERROR.  CANNOT CONVERT MATRIX OF KIND ',I5)
      ERROR = .TRUE.
      RETURN
      END

C ***** COMAND *****
      SUBROUTINE COMAND
      IMPLICIT REAL*8 (A-H,O-Z)
CC    READS THE COMMAND CARDS AND TAKES APPROPRIATE ACTION,
CC    EITHER DIRECTLY BY SKIPPING TO THE CORRESPONDING CODE IN
CC    COMAND, OR BY SETTING UP A CALL TO A SUBROUTINE
CC
      COMMON /MATCH2/ AA(30),BB(30),KBTS,SMINCL,
1      DWORD(10),DWORD2(10),DWORD3(10),DWORD7(7,10),MINCOM
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON /DERIVA/ G(30) ,G2(30)
      COMMON/COMMND/KDUM,CNAME(25),CNAM2(25),CNAM3(25),NNAME
      LOGICAL SMINCL
      INTEGER DWORD,DWORD2,DWORD3,CWORD,CWORD2,CWORD3
      INTEGER CNAME,CNAM2,CNAM3
      DIMENSION GF(30)
      DATA AGOOD,ABAD,ANONE/4HGOOD, 4H BAD, 4HNONE /
      IKARD = 0
50  IKARD = IKARD + 1
      CWORD = DWORD(IKARD)
      CWORD2=DWORD2(IKARD)
      CWORD3=DWORD3(IKARD)
      DO 51 I=1,7
51  WORD7(I)=DWORD7(I,IKARD)

```



```

      K = 1
      DO 60 I= 1, 7
        IF (WORD7(I)) 58,60,58
58 K = I + 1
60 CONTINUE
      IF (CWORD .EQ. CNAME(5)) K = 2
      KM1=K-1
      WRITE (ISYSWR,5001) IKARD,CWORD,CWORD2,CWORD3,
        (WORD7(I), I=1,KM1)
      WRITE (ISYSWR,5004)
      NFCNMX = WORD7(1) + 0.5D0
      IF (NFCNMX .LE. 0) NFCNMX = 1000
      EPSI = WORD7(2)
      IF (EPSI .LE. 0.D0) EPSI = 0.1D0 * UP
      NEWMIN = 0
      ITAUR = 0
      ISW(1) = 0
      DO 80 I= 1, NNAME
        IF (CWORD .EQ. CNAME(I)) GO TO 90
80 CONTINUE
      WRITE (ISYSWR,5005)
      STOP
90 GO TO (100,200,300,400,500,600,700,1100,900,1000,1100,1200,1300,
  * 1400,1500,1600,1700,1800,1402,2000,2100,2200,2300) , I
C      . . . . . NEW MINIMUM
95 WRITE (ISYSWR,5011)
      ITAUR = 0
      EPSI = 0.1D0*UP
C      . . . . . MINIMIZE
100 NF = NFCN
      CALL SIMPLX
      IF (ISW(1) .GE. 1) GO TO 50
      NFCNMX = NFCNMX + NF - NFCN
150 VTEST = 0.04D0
      GO TO 460
C      . . . . . SEEK
200 CALL SEEK
      GO TO 50
C      . . . . . SIMPLEX
300 CALL SIMPLX
      GO TO 50
C      . . . . . MIGRAD
400 VTEST = WORD7(3)
      IF (VTEST .LE. 0.0D0) VTEST = 0.01D0
460 NF = NFCN
      APSI = EPSI
      CALL MIGRAD
      IF (ISW(2) .GT. 2) GO TO 50
      IF (ISW(1) .EQ. 1) GO TO 50
      NFCNMX = NFCNMX + NF - NFCN
      NF = NFCN
      CALL SIMPLX
      IF (ISW(1) .EQ. 1) GO TO 50
      NFCNMX = NFCNMX + NF - NFCN
      CALL MIGRAD

```

```

      GO TO 50
C      . . . . . MINOS
500 IF (ISW(2) .LT. 1) GO TO 550
      EPSI = 0.1D0 * UP
      VTEST = 0.1D0
      CALL MINOS
      IF (NEWMIN .LT. 1) GO TO 50
      GO TO 95
550 WRITE (ISYSWR,5006)
555 WRITE (ISYSWR,5007)
      GO TO 50
C      . . . . . PUNCH
600 CALL MPUNCH
      GO TO 50
C      . . . . . PRINTOUT
700 ISW(5) = WORD7(1) + 0.5D0
      GO TO 50
C      . . . . . FIX
900 IT = WORD7(1) + 0.5D0
      IF (IT .LE. 0 .OR. IT .GT. NU) GO TO 905
      IF (LCORSP(IT) .LE. 0) GO TO 905
      CALL FIXPAR(IT, 0, ILAX)
      IF (ISW(2) .GT. 1) CALL MPRINT(1,AMIN)
      GO TO 50
905 WRITE (ISYSWR,5006)
      WRITE (ISYSWR,5012)
      GO TO 50
C      . . . . . RESTORE
1000 IT = WORD7(1) + 0.5D0
      CALL RESTOR(IT)
      GO TO 50
C      . . . . . END, EXIT
1100 IT = WORD7(1) + 0.5D0
      IF (ISW(4) .EQ. 1 .OR. IT .GT. 0) GO TO 1150
CSY   IFLAG = 3
CSY   WRITE (ISYSWR,5009)
CSY   CALL FCN(NPAR,G,F,U,IFLAG)
      NFCN = NFCN + 1
1150 IF (I .EQ. 11) STOP
      RETURN
C      . . . . . GRADIENT
1200 ISW(3) = 1
      IF (WORD7(1) .GT. 0.0D0) GO TO 50
      DO 1230 I= 1, NU
1230 GF(I) = 0.0D0
      CALL INTOEX(X)
      CALL FCN(NPAR,GF,AMIN,U,2)
      NFCN = NFCN + 1
      CALL DERIVE(GF,G2)
      ISW(3) = 0
      CALL DERIVE(G, G2)
      WRITE (ISYSWR,5013)
      ISW(3) = 1
      DO 1250 I= 1, NU
      LC = LCORSP(I)

```

```

      IF (LC .EQ. 0) GO TO 1250
      AWD = AGOOD
      IF (DABS(GF(LC)-G(LC)) .GT. DABS(G2(LC))) AWD = ABAD
      IF (GF(LC) .EQ. 0.D0) AWD = ANONE
      IF (AWD .NE. AGOOD) ISW(3) = 0
      WRITE(ISYSWR,5014) I,NAM(I),GF(LC),G(LC),G2(LC),AWD
1250 CONTINUE
      IF (ISW(3) .EQ. 0) WRITE (ISYSWR,5015)
      GO TO 50
C
      . . . . . CALL FCN
1300 IFLAG = WORD7(1)
      IF (IFLAG .EQ. 3) ISW(4) = 1
      CALL FCN(NPAR,G,F,U,IFLAG)
      NFCN = NFCN + 1
      IF(IFLAG.LE.5) GO TO 50
      CALL EXTOIN(X)
      CALL FCN(NPAR,G,AMIN,U,4)
      NFCN=NFCN+1
      IF (ISW(2) .LE. 1) GO TO 1350
      ISW(2) = 1
      WRITE (ISYSWR, 5010)
1350 CALL MPRINT(1,AMIN)
      GO TO 50
C
      . . . . . MATOUT
1400 IF(ISW(2).GE.2) GO TO 1405
C
      . . . . . HESSE
1402 CALL HESSE
      CALL MPRINT(1, AMIN)
1405 CALL MATOUT(0.0D0 , 1)
      GO TO 50
C
      . . . . . COVARIANCE
1500 NRAPE = WORD7(1) + 0.5D0
      IF (NRAPE .NE. NPAR) GO TO 1550
      READ (ISYSRD,5002) ((V(I,J),I=1,NRAPE),J=1,NRAPE)
      ISW(2) = 3
      CALL MATOUT(0.0D0 , 1)
      CALL MPRINT(1,AMIN)
      GO TO 50
1550 WRITE (ISYSWR,5006)
      WRITE (ISYSWR,5008)
      NRAP2 = NRAPE**2
      READ (ISYSRD,5002) (G(I),I=1,NRAP2)
      GO TO 50
C
      . . . . . IMPROVE
1600 CONTINUE
      IF (ISW(2) .LT. 2) GO TO 550
      CALL IMPROV
      IF (NEWMIN .EQ. 1) GO TO 150
      GO TO 50
C
      . . . . . ERROR DEF
1700 CONTINUE
      UP = WORD7(1)
      IF (UP .LE. 0.D0) UP = 1.0D0
      IF (ISW(2) .GE. 1) CALL MPRINT(1,AMIN)
      GO TO 50

```

```

C          . . . . . PAGE
1800 WRITE (ISYSWR,5005)
      GO TO 50

C          . . . . . CONTOUR
2000 CONTINUE
      IF (ISW(2) .LT. 1) GO TO 550
      KE1 = WORD7(1)
      KE2 = WORD7(2)
      IF (KE1 .LE. 0 .OR. KE2 .LE. 0) GO TO 905
      IF (KE1 .GT. NU .OR. KE2 .GT. NU) GO TO 905
      IF (LCORSP(KE1) .LE. 0 .OR. LCORSP(KE2) .LE. 0) GO TO 905
      NFCNMX = 1000
      CALL CONTOU
      CALL MPRINT(1,AMIN)
      IF (NEWMIN .LE. 0) GO TO 50
      GO TO 95

C          . . . . . STANDARD
2100 CONTINUE
C      CALL STAND
      GO TO 50

C          . . . . . RELEASE
2200 CONTINUE
      DO 2220 IRL=1,7
      KRL = WORD7(IRL)
      IF (KRL .EQ. 0) GO TO 50
      KRL = -IABS(KRL)
      CALL RESTOR(KRL)
2220 CONTINUE
      GO TO 50

C          . . . . . BLANK COMMAND CARD
2300 CONTINUE
      GO TO 50
5000 FORMAT (2A4, A2, 7F10.0)
5001 FORMAT (1H 10(1H*)/1H 3H*** I3,4H****,2A4,A2,7F15.5)
5002 FORMAT (7F10.0,10X)
5003 FORMAT (13E10.2)
5004 FORMAT (1H 10(1H*))
5005 FORMAT (1H1)
5006 FORMAT (1H+10X 17H(COMMAND IGNORED))
5007 FORMAT ( 118H0THE ABOVE COMMAND CANNOT BE EXECUTED BECAUSE COVARIA
1NCE MATRIX WAS NEITHER CALCULATED NOR SUPPLIED, OR WAS DESTROYED./
2/)
5008 FORMAT( 102H SIZE OF COVARIANCE MATRIX TO BE READ DOES NOT CORRESP
1OND TO NUMBER OF CURRENTLY VARIABLE PARAMETERS./)
5009 FORMAT ( 28H0CALL TO FCN WITH IFLAG = 3 /)
5010 FORMAT ( 51H NEW START POINT ASSUMED COVARIANCE MATRIX LOST)
5011 FORMAT ( 50H0NEW MINIMUM FOUND. GO BACK TO MINIMIZATION STEP./1H
1 ,      60(1H=)/60X1HV/60X1HV/60X1HV/57X7HVVVVVVV/58X5HVVVVV/59
2X      3HVVV/60X1HV///)
5012 FORMAT ( 91H0THE ABOVE COMMAND CANNOT BE EXECUTED BECAUSE IT REQU
1ESTS A PARAMETER THAT IS NOT VARIABLE.//)
5013 FORMAT ( 37H0CHECK OF GRADIENT CALCULATION IN FCN 12X 9HPARAME
1TER      6X 9HG(IN FCN) 3X 9HG(MINUIT) 4X 5HERROR 5X
2 9HAGREEMENT)
5014 FORMAT (45X I5,2X A10,3E12.4,4X A4)

```

```

5015 FORMAT ( 54H0MINUIT DOES NOT ACCEPT DERIVATIVE CALCULATIONS BY FC
1N)
END
C ***** CONTOU *****
SUBROUTINE CONTOU
IMPLICIT REAL*8 (A-H,O-Z)
CC      FINDS POINTS LYING ON CONTOURS OF A GIVEN FCN VALUE.
CC      AS A FUNCTION OF TWO VARIABLE PARAMETERS SPECIFIED BY THE
CC      CONTOUR COMMAND.  THE CONTOURS ARE THEN PLOTTED BY PLTCON.
CC
COMMON /CONPTS/ SPT(3,300), NSPT
COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
INCLUDE 'PAREXT.CCC'
COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
COMMON /VARIAN/ V(15,15)
COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
COMMON /VARIAT/ VT(15,15)
COMMON/CONST/PI02,PI,TPI,PI2,RADEG
COMMON /DERIVA/ G(30) ,G2(30)
DIMENSION DIG(5)
DATA ANEWMN,DIG / 1H*, 1H1, 1H2, 1H3, 1H4, 1H5 /
DATA NSTMX,THMAX,MINPT,TEN,NPTDIM / 10, 0.4D0, 10, 100.D0, 298 /
REAL*8 d01R360 / 360.D0 /
INTEGER d01BLANK/' '/
INTEGER CWORD,CWORD2,CWORD3
ALLOW(IEXT) = (U(IEXT)-ALIM(IEXT)) *(BLIM(IEXT)-U(IEXT))
KE1 = WORD7(1)
KE2 = WORD7(2)
NCON = WORD7(3)
IF (NCON .LE. 0) NCON = 2
IF (NCON .GT. 5) NCON=5
SPT(1,1) = U(KE1)
SPT(2,1) = U(KE2)
SPT(3,1) = d01BLANK
NSPT = 1
NEWMIN = 0
KI1 = LCORSP(KE1)
KI2 = LCORSP(KE2)
IF (ISW(2) .LT. 2) GO TO 40
CALL UCOPY(V,VT,MAXINT**2)
CALL VERMIN(VT,MAXINT,MAXINT,NPAR,IERR)
IF (IERR .LE. 0) GO TO 60
C      MATRIX NOT POSITIVE-DEFINITE
WRITE (ISYSWR,35)
35 FORMAT ( 40H0COVARIANCE MATRIX NOT POSITIVE-DEFINITE)
40 WRITE (ISYSWR,42)
42 FORMAT ( 26H0ONLY DIAGONAL ERRORS USED)
A11 = V(KI1,KI1) * UP
A22 = V(KI2,KI2) * UP
A12 = 0.0D0
GO TO 75

```



```

      IF (ISW(5) .GE. 2) WRITE (ISYSWR,900)
      THETB = PIO2 - A12/DSQRT(A11**2 + A12**2)
      THET1 = THETB
      R = 10.0D0 * RMAXAB
      XPT = U(KE1) - R*DCOS(THET1)
      YPT = U(KE2) - R*DSIN(THET1)
      SLOPE = 2.0D0 * (F2-AMIN) / (STEP2* DSIN(THET1))
      TOLER = TOLERN
      NSTEP = 0
      RMAX = RAVE
      GO TO 407

C      . . . . . CUT STEP LENGTH R
350 THET1 = 0.25D0 * (3.0D0*THETB + THET1)
      R = R * 0.5D0
      RMAX = R
      TOLER = TOLERN*R/RMAXAB
      SLOP = SLOPE
      NSTEP = 0

C      . . . . . FIND NEXT POINT
400 U(KE1) = XPT + R*DCOS(THET1)
      U(KE2) = YPT + R*DSIN(THET1)
      IF (ALLOW(KE1) .LT. 0.D0 .OR. ALLOW(KE2) .LT. 0.D0) GO TO 731
      CALL FCN(NPAR,G,F2,U,IFLAG)
      NFCN = NFCN + 1
      IF (F2 .LE. AMIN) GO TO 741

C      . . . . . CHECK IF TWO TRIES ARE NEEDED
      F1 = F2
      THET2 = THET1
      AFA = DABS(F2-AIM)
      IF (AFA .GT. 10.0D0*TOLER .AND. R .GT. RMIN) GO TO 350
      IF (AFA .LT. 0.2D0*TOLER) GO TO 420
407 DTHET = (F2-AIM) / (R*SLOPE)
      THMIN = 0.001D0*RAVE/R
      IF (DABS(DTHET) .GT. THMAX) DTHET = DSIGN(THMAX,DTHET)
      IF (DABS(DTHET) .LT. THMIN) DTHET = DSIGN(THMIN,DTHET)

C      . . . . . SUBSEQUENT TRIALS FOR NEXT PT
410 CONTINUE
      F1 = F2
      THET2 = THET1 + DTHET
      U(KE1) = XPT + R*DCOS(THET2)
      U(KE2) = YPT + R*DSIN(THET2)
      IF (ALLOW(KE1) .LT. 0.D0 .OR. ALLOW(KE2) .LT. 0.D0) GO TO 731
      CALL FCN(NPAR,G,F2,U,IFLAG)
      NFCN = NFCN + 1
      IF (F2 .LE. AMIN) GO TO 741
      SLOPE = (F1-F2)/(DTHET*R)
      IF (SLOPE .GT. 0.D0) GO TO 420
      IF (NPT .GT. 0) GO TO 415
      SLOPE = 2.0D0 * (AMIN-F2) / (DSIN(THET2) * (XMIN-U(KE1)))
      GO TO 420
415 IF (R .GT. RMIN) GO TO 350
      SLOPE = SLOP
420 CHANG = (F2-AIM)/(SLOPE*R)
      IF (DABS(CHANG) .GT. THMAX) CHANG = DSIGN(THMAX, CHANG)
      THETA = THET2 + CHANG

```

```

C          CHECK IF BEST TRIAL POINT IS ACCURATE ENOUGH
      DELF = DMIN1(DABS(F2-AIM), DABS(F1-AIM))
      IF (DELF .LT. TOLER) GO TO 430
      IF (NPT .EQ. 0) GO TO 425
      IF (NSTEP .EQ. 1 .AND. R .GT. RMIN) GO TO 350
425  NSTEP = NSTEP + 1
      IF (NSTEP .GT. NSTMX) GO TO 751
      THET1 = THET2
      DTHET = THETA - THET1
      GO TO 410

C          . . . . . ACCEPT NEW POINT
430  XPT = XPT + R*DCOS(THETA)
      YPT = YPT + R*DSIN(THETA)
      U(KE1) = XPT
      U(KE2) = YPT
      IF (ALLOW(KE1) .LT. 0.D0 .OR. ALLOW(KE2) .LT. 0.D0) GO TO 731
      THETAB = DABS(THETA-THETB)
      NSTEP = 0
440  IF (NPT .GT. 1) GO TO 445
      THBEG = THETA
      XBEG = XPT
      YBEG = YPT
445  CONTINUE
      NPT = NPT + 1
      NSPT = NSPT + 1
      SPT(1,NSPT) = XPT
      SPT(2,NSPT) = YPT
      SPT(3,NSPT) = SYM
      XPM = XPT - XMIN
      YPM = YPT - YMIN
      IF (ISW(5) .LT. 2) GO TO 450
      THDEG = THETA/RADEG
      THDEG = DMOD(THDEG, 360)
      WRITE (ISYSWR, 901) NPT, XPT, YPT, XPM, YPM, DELF, R, THDEG, NFCN
450  CONTINUE

C          . . . . . TEST IF CONTOUR COMPLETED
      IF (NPT .LT. MINPT) GO TO 500
      IF ((YPT-YBEG)**2 + (XPT-XBEG)**2 .GT. R**2) GO TO 500
      IF (DCOS(THETA-THBEG) .GT. 0.5D0) GO TO 600
500  IF (NPT .GE. NPTMX) GO TO 761

C          . . . . . ESTIMATE NEW POINT
      RINV = THETAB**2 / (R*DCA**2)
      IF (RINV .GT. 1.0D0/RMAX) GO TO 522
      R2 = RMAX
      GO TO 540
522  R2 = 1.0D0/RINV
      IF (R2 .LT. RMIN) R2 = RMIN
540  CHANG = R2*RINV*DCA
      IF (CHANG .GT. THMAX) CHANG = THMAX
      IF (THETB .GT. THETA) CHANG = -CHANG
      THET1 = THETA + CHANG
      THETB = THETA
      R = R2
      RMAX = DMIN1(2.0D0*R, RMAXAB)
      TOLER = TOLERN * R/RMAXAB

```


SLOP = SLOPE

GO TO 400

C

CONTOUR COMPLETED NORMALLY

600 WRITE (ISYSWR,605) SYM,NPT

605 FORMAT (9H0CONTOUR A3, 18H IS COMPLETED WITH15, 7H POINTS//)

GO TO 790

C

ERROR RETURNS

731 WRITE (ISYSWR, 735) U(KE1), U(KE2)

735 FORMAT (34H0CONTOUR LEAVING ALLOWED REGION AT2E16.7)

GO TO 780

741 WRITE (ISYSWR, 745)

745 FORMAT (38H0NEW ABSOLUTE MINIMUM FOUND BY CONTOUR/)

NEWMIN = 1

NSPT = NSPT + 1

SPT(1,NSPT) = U(KE1)

SPT(2,NSPT) = U(KE2)

SPT(3,NSPT) = ANEWMN

AMIN = F2

CALL EXTGIN(X)

CALL MPRINT(0,AMIN)

GO TO 810

751 WRITE (ISYSWR, 755)

755 FORMAT (39H0TOO MANY ITERATIONS TO FIND NEXT POINT)

GO TO 780

761 WRITE (ISYSWR, 765)

765 FORMAT (45H0MEMORY OVERFLOW. TOO MANY POINTS IN CONTOUR)

780 WRITE (ISYSWR,1001) NPT

790 U(KE1) = XMIN

U(KE2) = YMIN

800 CONTINUE

C

END MAIN LOOP

810 CONTINUE

WRITE (ISYSWR,1002) KE1,KE2,UP,KE2

CALL PLTCON(NSPT,SPT)

WRITE (ISYSWR,1003) KE1

RETURN

900 FORMAT (8X 2HPT 9X 1HX 13X 1HY 10X 6HX-XMIN 6X 6HY
1-YMIN 7X 8HACCURACY 4X 10HSEPARATION 5X 5HTHETA 3X 5HCA
2LLS)

901 FORMAT (I10,2F14.7,2X,2F12.7,F13.6,2X, F12.7,F10.1,I8)

1000 FORMAT (15H0PARAMETERS NO.14, 4H AND14,10X 8HCONTOUR A3, 12
1H F = FMIN +F11.6)

1001 FORMAT (55H CONTOUR NOT COMPLETED BECAUSE OF ABOVE ERROR CONDITI
1ON/1X 54(1H*),15, 43H POINTS ON THE CONTOUR HAVE BEEN DETERMINED/
2/)

1002 FORMAT(36H1 FUNCTION CONTOURS FOR PARAMETERS 214, 20X,

1 20H CONTOUR 1 = FMIN + F10.5/12H0 PARAMETER 13)

1003 FORMAT (25H+ PARAMETER 13,1H./)

END

C..... CYAE

SUBROUTINE CYAE(M)

IMPLICIT REAL*8 (A-H,O-Z)

USING BETATRON FUNCTIONS, COMPUTES BEAM ENVELOPES ALONG BEAM

- * LINE FROM PREVIOUSLY CALCULATED EMITTANCES, EPSX,EPY,DP/P.

C.....

- * NAME CYAE KA /BL BEAM KSQ EPXCO EPYCO

C.....

- * BL = BEAMLINE NAME
- * BEAM = NAME OF BVAL OR CYEM THAT GIVES EMITTANCES AND DP/P
- * KSQ = EPS/EPY0, WHERE EPS0 IS EMITTANCE FROM BVAL (SIG**2/BETA)
- * AND EPS IS EMITTANCE TO BE USED BY CYAE.
- * XB AND XP ARE THUS MULTIPLIED BY SQRT(KSQ).
- * EPXCO, EPYCO = EQUIVALENT EMITTANCES THAT GIVE CLOSED ORBIT DISPLACEMENTS
- * KA = 0 OR NON-0 — ADD BETATRON AND MOMENTUM DISPLACEMENTS IN QUADRATURE
- * OR ALGEBRAICALLY.

INCLUDE 'BINFF.CCC'

INCLUDE 'BMI.CCC'

INCLUDE 'DIM.CCC'

COMMON/MC/ML

DIMENSION P(4),BETX(9),BETY(9),ML(3)

EQUIVALENCE (EPX,P(1)),(EPY,P(2)),(EPL,P(3)),(DPP,P(4))

EQUIVALENCE (BX,BETX(2)),(GX,BETX(4)),(DX,BETX(5)),(DDX,BETX(6)),

1 (BY,BETY(2)),(GY,BETY(4)),(DY,BETY(5)),(DDY,BETY(6))

DATA IBLNK/1H /

- * GET DATA EPX,EPY AND DP/P

MD = MDAT(M,2)

CALL DATA (MD,5,1,4,P)

- * KA DETERMINES CALCULATION FOR XTOT AND XTOT PR

KA = INFF(4,M)

- * GET BEAM LINE, MAKE MLIST AND MULTIPLY MATRICES

- * K IS NUMBER OF ELEMENTS IN BEAM

MB = MDAT(M,1)

K = INFF(17,MB)

MS = MEND - 1

CALL MIFILL (MB,1,K,NELS,MI)

CALL MIM (MS,NELS,MI)

KW = NELS + 1

MI(KW) = MI(1)

I = 0

EL = 0.

FL = 0.

MAT = IBLNK

XKSQ=FDAT(M,1)

RK=DSQRT(XKSQ)

EPXCO=FDAT(M,2)

EPYCO=FDAT(M,3)

RX=DSQRT(EPXCO/EPX)

RY=DSQRT(EPYCO/EPY)

```

*   PRINT TABLE HEADING AND SIGX AND SIGY, AS DEFINED BY FACTK
      CALL DASH
      WRITE(3,1) P(1),EPXCO,P(3),P(2),EPYCO,P(4)
      IG = 1
      IF (KA.EQ.0) IG = 2
      GO TO (104,105) IG
104  WRITE (3,6)
      GO TO 106
105  WRITE (3,8)
106  WRITE(3,2) RK
      WRITE(3,11)

115  CONTINUE
      CALL BET(MS,BETX,BETY,1)
*   COMPUTE AND PRINT BEAM ENVELOPES
      X=DSQRT(EPX*BX)*RK
      Y=DSQRT(EPY*BY)*RK
      XP=DX*DPP
      YP=DY*DPP
      XCO=RX*X
      YCO=RY*Y
      XPR=DSQRT(EPX*GX)*RK
      YPR=DSQRT(EPY*GY)*RK
      XPPR=DDX*DPP*RK
      YPPR=DDY*DPP*RK
      XCOPR=RX*XPR
      YCOPR=RY*YPR
      GO TO (116,117) IG
116  XTOT = X + DABS(XP)
      XPRTOT = XPR + DABS(XPPR)
      YTOT = Y + DABS(YP)
      YPRTOT = YPR + DABS(YPPR)
      GO TO 118
117  CONTINUE
      XTOT = DSQRT( X * X + XP * XP )
      XPRTOT = SQRT( XPR*XPR + XPPR*XPPR)
      YTOT = DSQRT(Y*Y + YP*YP)
      YPRTOT = DSQRT(YPR*YPR + YPPR*YPPR)
118  CONTINUE
      XTOT=XTOT+XCO
      YTOT=YTOT+YCO
      XPRTOT=XPRTOT+XCOPR
      YPRTOT=YPRTOT+YCOPR
      EL = EL + FL
      WRITE(3,12) I,MAT,EL,X,XP,XCO,XTOT,Y,YP,YCO,YTOT,XPRTOT,YPRTOT

      I = I + 1
      IF (I.EQ.KW) GO TO 125
      NI = MI(I)
      CALL DATA(NI,5,1,1,FL)
      MAT = MNAME(NI)
      ML(1) = - NI
      ML(2) = MS
      ML(3) = NI

```

```
CALL MMM(MS,3,ML)
GO TO 115
```

```
125 CALL DASH
RETURN
```

```
.....

1      FORMAT (3X,'BEAM ENVELOPES (MM,MRAD)'/
1 5X,'EMITTANCES (MM-MRAD) — EPSX = ',F10.6,5X,'EPSXCO = ',
2 F10.6,5X,'EPSL = ',F13.6/30X,'EPSY = ',F10.6,5X,'EPSYCO = ',
3 F10.6,5X,'SIGP = ',F10.6,' (0/00)')

6      FORMAT (5X,'XTOT = XB + XP + XCO, ETC.')
8      FORMAT (5X,'XTOT = SQRT(XB*XB + XP*XP) + XCO, ETC.')

2      FORMAT(1H+,99X,'DISPLACEMENT = ',F5.2,1H*, 'SIGMA')
9      FORMAT (I4,1X,A5,F9.4,12F9.4)
10     FORMAT (/2X,'POS',10X,'S',8X,'X',7X,'X PR',6X,
1 'XP',5X,'XP PR',4X,'XTOT',3X,'XTOT PR',5X,'Y',7X,
2 'Y PR',5X,'YP',6X,'YP PR',5X,'YTOT',3X,'YTOT PR'/1X,131(1H-))
11     FORMAT(/2X,'POS',10X,'S',13X,'XB',7X,'XP',7X,'XCO',6X,'XTOT',
1 10X,'YB',7X,'YP',7X,'YCO',6X,'YTOT',
2 10X,'XPRTOT',3X,'YPRTOT'/1X,131(1H-))
12     FORMAT(I4,1X,A5,F9.4,5X,4F9.4,5X,4F9.4,5X,2F9.4)
```

```
END
```

```
C ***** CYCLE *****
SUBROUTINE CYCLE(M,K,MI,L,WS)
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'DIM.CCC'
DIMENSION M3(3)
DIMENSION MI(1 )
INTEGER WS
IF (WS.EQ.0) GO TO 5
NO=MEND-3
NC=K
GO TO 6
5 NO=M-1
NC=K-1
6 IF (L.EQ.0) GO TO 1
CALL MMM(M,K,MI)
1 MI( K+1)=MI(1)
DO 2 I=1,NC
IF (I.NE.1) GO TO 11
M3(2)=M-1
NJ=NO - 1
GO TO 12
11 NJ=NO - I
M3(2) = NJ + 1
12 M3(3) = MI( I )
M3(1) =-M3(3)
CALL MMM (NJ,3,M3)
2 CONTINUE
RETURN
```

```

      END
C ***** CYEM *****
      SUBROUTINE CYEM(M)
      IMPLICIT REAL*8 (A-H,O-Z)
C  CALCULATES ELECTRON INTEGRALS, RF OUTPUT, ELECTRON EMITTANCE FACTORS
C
C  EPS  CYEM  KA  KB  //  BL  E  R  F  V  K
C
C  KA = OUTPUT OPTION
C  KB = NUMBER OF SUPERPERIODS
C      IF BL IS A FXPT INSTRUCTION, BEAM LINE AND KB ARE TAKEN FROM
C      THOSE OF THE FXPT INSTRUCTION.
C  BL = NAME OF BEAM LINE (BML INSTRUCTION) OR OF FXPT INSTRUCTION
C  E = ELECTRON BEAM ENERGY (GEV)
C  R = MACHINE RADIUS (M)
C  F = RF FREQUENCY (MHZ)
C  V = RF VOLTS/TURN (MV)
C  K = COUPLING COEFFICIENT (SQRT(EPY/EPX))
C  KA = OUTPUT OPTION. IGNORED (SET TO 0) IF BL IS A BML INST
C  KA = 0,1 PRINT ONLY AT START OF BEAM LINE
C  KA = 2,3,4 PRINT AT EVERY ELEMENT
C  KA = EVEN, BETATRON PART OF SIGMA MATRIX
C  KA = ODD, TOTAL SIGMA MATRIX
C  KA = 2,3 SIGMA MATRIX LONG OUTPUT
C  KA = 4 AIJ, BIJ LONG OUTPUT
C
C  KA POSITIVE — PRINTS RII, TIJ. RII = SQRT(SIGMA(I,I))
C                  TIJ = ARCTAN(SIJ/SII - SJJ) / 2
C                  RII IS THE PROJECTION OF BEAM ELLIPSE ON I AXIS
C                  TIJ IS TILT OF ELLIPSE PROJECTION IN I-J PLANE
C  KA NEGATIVE — PRINTS SIJ = SIGMA(I, J) FOR ALL ELEMENTS
C                  AIJ = 4 X REAL(F1I* X F1J),BIJ = 4 X REAL(F3I* X F3J)
C                  F=INVERSE(E),F=MATRIX OF EIGENVECTORS,FIJ*X CONJ(FIJ)
C                  SIGMA = INVERSE(A/EP1 + B/EP3)
C                  EP1, EP3 = GENERALIZED EMITTANCES (SIGMA**2/BETA)
C                  STOTIJ = SBETIJ + ETAJ*SIGDP**2
C                  IF EP3 = 0, SETS EP3 = EPY
C                  IF KAPPA = 0 AND BL = FXPT INSTR, SETS KAPPA=E-04
C
      INCLUDE 'STORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (ISTORE(1),STORE(1))
      INCLUDE 'BML.CCC'
      INCLUDE 'DIM.CCC'
      INCLUDE 'CONST.CCC'
      common/seed/nr

      DIMENSION BETX(9),BETY(9),PAR(5),P(6),EM(4),ML(3)
      DIMENSION EV(4,8),VZ(7,7),VW(7),RW(3),AA(4,4),BB(4,4),CC(4,8)
      DIMENSION RX(2,3),RY(2,3)
      DIMENSION EI(2)
      COMPLEX*16 EV,DET,EJ1,EJ3,EK1,EK3,EI,XM2I

```

```

REAL*8 I41,I43,I51,I53,J1,J3,J6,K,KSQ,LH
EQUIVALENCE (L,P(1)),(DB,P(2)),(BRHO,P(3)),(BZ,P(4)),(OM,P(4))
EQUIVALENCE (GAM,BETX(4))
INTEGER BDAT

```

```

EQUIVALENCE (PSI,XI)
EQUIVALENCE (EPX,EM(1)),(EPY,EM(2)),(DPP,EM(3)),(EPLN, EM(4))
EQUIVALENCE (B,BETX(2)),(A,BETX(3)),(E,BETX(5)),(F,BETX(6))
EQUIVALENCE (PC,PAR(1)),(R,PAR(2)),(FREQ,PAR(3)),(V0,PAR(4)),
1 (FKAP,PAR(5))
EQUIVALENCE (ANG0,IANG0), (ANG1,IANG1)

```

```

REAL*8 I1,I2,I3,I4,I5,L,KX,KK,N,LMTOT,K3R,JE,JX,JZ
LOGICAL BSW, LONG,FLAG,SIGFLG
DATA RME/.0005110041D0/
DATA CGAM,CQP,C,JE,JX,JZ
1/8.846002206D-5,1.467471313D-6,2.997925D8,2.,1.,1./
DATA RW/3*0.0/
INTEGER doIMAG/'MAG '/,doIBLANK/' '/,doIDOL/'$ '/
INTEGER doIMOVE/'MOVE'/,doIFXPT/'FXPT'/

```

```

nrn=nr

```

```

C GET DATA P/C,R,FREQ,VRF,KAPPA
C UNITS — GEV/C, METERS, MHZ, MV
CALL DATA (M,1,1,5,PAR)
FREQ = FREQ*1.D6
V0 = V0*1.D-3

C GET BEAM LINE, MAKE MI LIST AND MULTIPLY MATRICES
C K IS NUMBER OF ELEMENTS IN BEAM LINE
MB = MDAT(M,1)
KA = 0
KB = INFF(5,M)
FLAG = .FALSE.
MS = MEND - 1
IOPMB = INFF(1,MB)
NGO=1
IF (IOPMB.EQ.doIFXPT) NGO=2
CALL DASH
WRITE (3,13)
GO TO (51,52), NGO
52 MFX=MB
MB=MDAT(MFX,2)
NP=MDAT(MFX,1)
KA = INFF(4,M)
SIGFLG = .FALSE.
IF (KA .GE. 0) GO TO 55
SIGFLG = .TRUE.
KA = -KA
55 CONTINUE
KAF = INFF(4,MFX)
LONG = .FALSE.
IF (KA.GT.1) LONG = .TRUE.
KB = INFF(5, MFX)
MS7=M7END-3

```

```

53  CONTINUE
    CALL DATA(MFX,7,1,49,VZ)
    CALL STOR7(MS7,VZ,RW)
    CALL DATA(MFX,7,1,7,VW)
    CALL RXY(MFX,RX,RY,RW)
    CALL STXY(MS,RX,RY,RW)
    IF (.NOT.FLAG) GO TO 54
    II = 0
    CALL DASH
    IF (KA .EQ. 1 .OR. KA .EQ. 3) GO TO 56
    WRITE(3, 19)
    GO TO 57
56  CONTINUE
    WRITE(3, 20)
57  CONTINUE
    IF (SIGFLG) GO TO 58
    WRITE (3,16)
    GO TO 503
58  CONTINUE
    WRITE(3, 18)
    GO TO 503
54  CONTINUE
    WRITE(3,14)
51  CONTINUE
    I = INFF(17,MB)
    CALL MIFILL (MB,1,I,NELS,MI)
    IF (NGO.EQ.2) GO TO 59
    CALL MMM(MS,NELS,MI)
    IF (KB .GE. 0) GO TO 59
    NELS = NELS +1
    MI(NELS) = MEND -6
    CALL REF(MEND-6, MS)
    ML(1) = MS
    ML(2) = MEND -6
    CALL MMM(MS, 2, ML)
59  CONTINUE
    IF (KB .LT. 0) KB = -2 * KB
    KW = NELS + 1
    MI(KW) = MI(1)
    BSW = .FALSE.
    I1 = 0.
    I2 = 0.
    I3 = 0.
    I4 = 0.
    I5 = 0.
    I41 = 0.
    I43 = 0.
    I51 = 0.
    I53 = 0.
    II = 0

```

C*****c*****

```

100 CALL BET(MS,BETX,BETY,1)
    IF (II.EQ.0) BX0=BETX(2)

```

```

      IF (NGO.EQ.2) GO TO 503
101  II = II + 1
C    IF I = KW, END OF BEAM LINE
      IF (II.GT.KW) GO TO 400
      NI = MI(II)
      IF (FLAG) GO TO 121
C    IF NOT AT BEGINNING OR END OF A BENDING MAGNET, CONTINUE
110  IOP = INFF(1,NI)
      IF (BSW) GO TO 300
      IF (IOP.NE.doIMOVE) GO TO 115
      MQ = MDAT(NI,1)
      IOP = INFF(1,MQ)
      IF (IOP.NE.doIMAG) GO TO 120
      CALL DATA(MQ,1,1,6,P)
      GO TO 116
115  IF (IOP.NE.doIMAG) GO TO 120
C      GET MAGNET DATA.  L,K,RHO,OMEGA,GAMMA,DELTA
      CALL DATA (NI,1,1,6,P)
116  IF ( (BZ.NE.0.) .OR. (NGO.EQ.2) ) GO TO 200
120  CONTINUE

      GO TO (122,121), NGO
121  NF = M7END - 5
      CALL TRKCN(MS7,NI,VW,NP,0)
      NI = NF
      IF (KAF.EQ.0) GO TO 122

C  PRINT AIJ, BIB
900  CONTINUE
      IPOS = II-1
      IF(IPOS.NE.0) GO TO 901
      MATI=doIBLANK
      IF (FLAG) GO TO 903
      GO TO 902
901  IF (.NOT.LONG) GO TO 122
      MK=MI(IPOS)
      MATI=INFF(2,MK)
      IF (FLAG) GO TO 903
      IF (KA .NE. 4) GO TO 122
902  CONTINUE
      WRITE(3, 15) IPOS, MATI,
1    AA(1,1),AA(1,2),AA(2,2),AA(1,3),AA(1,4),
2    AA(2,3),AA(2,4),AA(3,3),AA(3,4),AA(4,4),
3    BB(1,1),BB(1,2),BB(2,2),BB(1,3),BB(1,4),
4    BB(2,3),BB(2,4),BB(3,3),BB(3,4),BB(4,4)
      GO TO 122
903  CONTINUE
      WRITE(3, 17) IPOS, MATI,
1          CC(1,1),CC(1,2),CC(2,2),CC(1,3),CC(1,4),
2          CC(2,3),CC(2,4),CC(3,3),CC(3,4),CC(4,4)
      IF (.NOT. LONG .OR. KA .EQ. 4) GO TO 400
      GO TO 503

122  CONTINUE
      ML(1) = -NI

```



```

ML(2) = MS
ML(3) = NI
CALL MMM(MS,3,ML)
GO TO 100

```

C*****

C CALCULATE BILINEAR FORMS — GENERALIZED ELLIPSE MATRICES

```

503 CALL RTRV7(MS7,VZ,RW)
    H = GAM*E*E + 2.*A*E*F + B*F*F
    ASSIGN 504 TO IRET
500 DO 501 I=1,4
    EV(I,1) = DCMLPX( VZ(I,2),VZ(I,3) )
    EV(I,3) = DCMLPX( VZ(I,4),VZ(I,5) )
    EV(I,2) = DCONJG( EV(I,1) )
    EV(I,4) = DCONJG( EV(I,3) )
    DO 502 J=5,8
502 EV(I,J) = DCMLPX(0.D0,0.D0)
501 EV(I,I+4) = DCMLPX(1.D0,0.D0)
600 DO 601 J=1,2
    I=2*J-1
601 EI(J) = DCONJG( VZ(I,6)*EV(2,I) - VZ(2,6)*EV(1,I)
1      + VZ(3,6)*EV(4,I) - VZ(4,6)*EV(3,I) )
    EJ1 = EI(1)*EV(1,1)
    EJ3 = EI(1)*EV(3,1)
    EK1 = EI(2)*EV(1,3)
    EK3 = EI(2)*EV(3,3)
    CALL CMXDIV(EV,4,4,DET,4)
    GO TO IRET,(53,504)
504 DO 505 I=1,4
    DO 505 J=1,4
    AA(I,J) = 4.0 * DREAL( DCONJG(EV(1,I+4)) * EV(1,J+4) )
    BB(I,J) = 4.0 * DREAL( DCONJG(EV(3,I+4)) * EV(3,J+4) )
    IF (.NOT.FLAG) GO TO 505
    CC(I,J) = REP1*AA(I,J) + REP3*BB(I,J)
    UU = 0.
    IF (J.EQ.I) UU = 1.
    CC(I,J+4) = UU
505 CONTINUE
    IF (.NOT. FLAG) GO TO 101
    CALL MXDIV(CC, 4, 4, DCC, 4)
    IF (.NOT. SIGFLG) GO TO 510
    DO 511 I = 1,4
    DO 511 J = 1,4
        CC(I,J) = CC(I, J+4)
511 CONTINUE
    GO TO 101
510 CONTINUE
    DO 506 I = 1,4
    DO 506 J = 1,4
        SPII = 0.
        SPJJ = 0.
        SPIJ = 0.
        IF (KA .NE. 0 .AND. KA .NE. 3) GO TO 509
        SPII = VZ(I,6)*VZ(I,6)*SIGPSQ

```

```

      SPJJ = VZ(J,6)*VZ(J,6)*SIGPSQ
      SPIJ = VZ(I,6)*VZ(J,6)*SIGPSQ
509  CONTINUE
      SII = CC(I, I+4) + SPII
      IF (J .EQ. I) GO TO 507
      SIJ = CC(I, J+4) + SPIJ
      SJJ = CC(J, J+4) + SPJJ
      YY = 2. * SIJ
      XX = SII - SJJ
      TIJ = 0.
      IF (YY .EQ. 0 .AND. XX .EQ. 0) GO TO 508
      TIJ = DATAN2(YY, XX) / 2.
508  CONTINUE
      CC(I, J) = TIJ
      GO TO 506
507  CONTINUE
      CC(I, I) = DSQRT(SII)
506  CONTINUE
      GO TO 101

```

C BENDING MAGNET — ENTRANCE

```

200  LH = L/2.
      BR = BRHO*(1. + VW(6))
      IF (BZ.EQ.0.) GO TO 262
      G = BZ/BR
      RHO = 1./G
      KX = P(2) + OM*G
      KK = DSQRT(DABS(KX/P(3)))
      PHI = KK*L
      N = - P(2) * P(3) / (OM*OM)
      G2 = G*G
      G3 = G2*G
      AG3 = DABS(G3)
      AG3L = AG3*LH
      DFAC = G3*(1.-2.*N)
      EL2G = DFAC*L*L*G/24.D0
      Q10 = DFAC*LH
      Q3 = 0.
      THET = L * DABS(G)

```

C E0,E1 ARE ENTRANCE EXIT ANGLES

```

      ANG0=P(5)
      ANG1=P(6)
      IF ( (IANG0.EQ.dol dol).OR.(IANG1.EQ.dol dol) ) GO TO 201
      E0 = P(5)*RADEG
      E1 = P(6)*RADEG
      GO TO 203
201  E0 = .5*THET
      E1=E0
203  CONTINUE
      TE0 = DTAN(E0)
      TE1 = DTAN(E1)
      GAM =(1. + A*A)/ B

      IF (E0.EQ.0.) GO TO 202

```

```

F = F + E*TE0/RHO
A = A - B*TE0/RHO
GAM = (1. + A*A)/B
202 K3R = 1. / (KK*KK*KK*RHO)

```

```

IF (KX) 230,210,220

```

```

C          COMPUTE V. DO U AT END OF MAGNET

```

```

C          FOR KX = 0

```

```

210 CONTINUE
RH02 = RHO*RHO
XL2=L*L
XL3=L*XL2
XL4=L*XL3
XL5=L*XL4
V = ( GAM*E*E + 2.*A*E*F + B*F*F ) * L
1  + ( A*E + B*F ) * XL2 / RHO
2  + ( B/RHO - GAM*E - A*F ) * XL3 / ( 3.*RHO )
4  - A * XL4 / ( 4.*RHO2 ) + GAM * XL5 / ( 20.D0*RHO2 )
GO TO 250

```

```

C          FOR KX.GT.0, F MAGNET

```

```

220 CONTINUE
T = DTAN(.5*PHI)
CS = DCOS(PHI)
S = DSIN(PHI)
SC = S*CS
VC1 = 1. - CS
VC2 = PHI - S
VC3 = 2.*K3R
VC4 = 1./(KK*B)
V1 = L * (GAM*E*E + 2.*A*E*F + B*F*F)
V2 = VC3*E * (KK*A*VC1 - GAM*VC2)
V3 = VC3*F * (KK*B*VC1 - A*VC2)
VC5 = .5*(PHI-SC) -2.*A*VC4*(VC1 - .5*S*S)
VC6 = 1.5D0*PHI - 2.*S + .5*SC
V4 = (B*K3R / RHO) * (VC5 + GAM*VC4*VC6/KK)
V = V1 + V2 + V3 + V4
GO TO 250

```

```

C          FOR KX.LT.0 D MAGNET

```

```

230 CONTINUE
EX = DEXP(PHI)
EXM = DEXP(-PHI)
SH = .5*(EX - EXM)
CH = .5 * (EX + EXM)
SC = SH*CH
VC1 = 1. - CH
VC2 = PHI - SH
VC3 = 2.*K3R
VC4 = 1./(KK*B)
VC5 = PHI - SC
VC6 = VC1 + .5*SH*SH
VC7 = 1.5D0*PHI - 2.*SH + .5*SC

```

```

V1 = L* (GAM*E*E + 2.*A*E*F + B*F*F)
V2 = VC3*E*(KK*A*VC1 - GAM*VC2)
V3 = VC3*F* (KK*B*VC1 - A*VC2)
V4 = (B*K3R/RHO) * (.5*VC5 + 2.*A*VC4*VC6 - GAM*VC4*VC7/KK)
V = V1 - V2 - V3 - V4

```

GO TO 250

C QUADRUPOLE

```

262 K = DB/BR
    EL2G = 0.
    KSQ = K*K
    XK2L = KSQ*LH
    AK3L = DABS(K*KSQ*LH)
    TKSQ = 2.*KSQ*LH
    W1=0.
    GO TO 260

250 GO TO (251,260), NGO
260 IF (BZ.NE.0.) GO TO 264
    RSQ = VW(1)*VW(1) + VW(3)*VW(3)
    RQ = DSQRT(RSQ)
    G2L = RSQ*XK2L
    AG3L = RQ*RSQ*AK3L
    I2 = I2 + G2L
    I3 = I3 + AG3L
    Q1 = TKSQ*VW(1)
    Q3 = TKSQ*VW(3)
    GO TO 263

264 Q1 = Q10 - TE0*G2
    IF (BSW) Q1 = Q10 - TE1*G2

263 I41 = I41 - Q1*DIMAG(EJ1) - Q3*DIMAG(EJ3) - EL2G*L
    I43 = I43 - Q1*DIMAG(EK1) - Q3*DIMAG(EK3)
    DO 261 I=1,4
    DO 261 J=1,4
    W1 = W1 + AA(I,J)*VZ(I,6)*VZ(J,6)*AG3L
261 I53=I53 + BB(I,J)*VZ(I,6)*VZ(J,6)*AG3L
    W1 = W1 - H*AG3L
    IF (BSW) GO TO 302

```

```

251 CONTINUE
    BSW = .TRUE.

```

C SAVE E TO COMPUTE U AT END OF MAGNET

```

    ESAV = E
    GO TO 120

```

C BENDING MAGNET EXIT. CALCULATE U AND DO SUMMING.

```

300 CONTINUE
    GO TO (301, 260), NGO
302 IF (OM.EQ.0.) GO TO 550
301 IF (KX) 330,310,320
310 U = (ESAV + E) * .5*L - (L*L*L)/(12.D0*RHO)
    GO TO 340
320 T = DTAN(.5*PHI)

```

```

      U = (ESAV + E) * T / KK + K3R * (PHI - 2. * T)
      GO TO 340
330   T = DTANH(.5 * PHI)
      U = (ESAV + E) * T / KK - K3R * (PHI - 2. * T)
340   CONTINUE
      I1 = I1 + G * U
      I2 = I2 + G * G * L
      I3 = I3 + AG3 * L
      TR = (ESAV * DTAN(E0) + E * DTAN(E1)) / (RHO * RHO)
      I4 = I4 + DFAC * U - TR
      I5 = I5 + AG3 * V
550   IF (NGO.EQ.2) I51=I5+W1
      BSW = .FALSE.
C     GO BACK TO FIND OUT IF THIS ALSO BEGINS A BENDING MAGNET.
      GO TO 115

```

C.....

```

C           RF OUTPUT
400   CONTINUE
      nr=nrn
      IF (NGO .EQ. 1) GO TO 561
      IF (FLAG) GO TO 563
      FLAG = .TRUE.
561   CONTINUE
      I1 = I1 * KB
      I2 = I2 * KB
      I3 = I3 * KB
      I4 = I4 * KB
      I5 = I5 * KB
      GO TO (552,553), NGO
553   I51 = I51 * KB
      I53 = I53 * KB
      I41 = I41 * KB
      I43 = I43 * KB
      D1 = I41 / I2
      D3 = I43 / I2
      J1 = 1.0 - D1
      J3 = 1.0 - D3
      J6 = 2.0 + D1 + D3
552   CONTINUE
      LMTOT = PI2 * RHO
      EL = PI2 * R
      PC4 = PC ** 4
      ALPHA = I1 / EL
      GAMT = DSQRT(1. / ALPHA)
      U0 = CGAM * PC4 * I2 / PI2
      D = I4 / I2
      JX = 1.0 - D
      JE = 2.0 + D
      T0 = EL / C
      TAU0 = 4. * PI * T0 / (CGAM * PC ** 3)
      TAUy = TAU0 / I2
      TAUx = TAU0 / (I2 - I4)
      TAUe = TAU0 / (2. * I2 + I4)

```

```

SQSIG= DABS(CQP*PC*13/(JE*12))
SIGE = DSQRT(SQSIG)
FK = FREQ*T0
CPHIS = U0/V0
PHIS = DARCOS(CPHIS)
TE = DSQRT(PI2*PC*T0*T0 / (ALPHA*FK*V0*DSIN(PHIS)))
SQE = 2.*U0*(DTAN(PHIS) - PHIS)/(PI*FK*ALPHA*PC)
ASQE = DABS(SQE)
EPSM = PC*DSQRT(ASQE)
PSI = (EPSM/SIGE)**2/2.
IF (PSI .LE. 70.0D0) GO TO 556
PSITRU = PSI
PSI = 70.0D0
WRITE(3, 1) PSITRU, PSI
556 CONTINUE
TAUQ =(TAUE/2.)*DEXP(PSI) /PSI
SIGPHI = SIGE*DSQRT (PI2*FK*ALPHA / (V0*DSIN(PHIS)*PC))
SIGL = SIGPHI*R/FK
PHISD = PHIS*180.D0/PI

```

C COMPUTE EPSILONS FOR EMITTANCE AND STORE

```

BETX(2)=BX0
SIGXOB = DSQRT (CQP*PC*PC*15 / (JX*12))
SIGX = SIGXOB * DSQRT(BETX(2))
SQRT6 = DSQRT(6.D0)
EPX0 = SIGX*SIGX/BETX(2)
IF (FLAG .AND. FKAP .EQ. 0.) FKAP = .0001D0
EPX = (1. - FKAP*FKAP)*EPX0
EPY = FKAP*FKAP*EPX0
GO TO (554,555), NGO
555 CK = CQP*PC*PC/12
EP1 = CK*151/J1
EP3 = CK*153/J3
EP3S = EP3
IF (EP3 .EQ. 0.) EP3S = EPY
REP1 = 1. / EP1
REP3 = 1. / EP3S
554 CONTINUE
SIGBG = SIGE/RME
EPLN = SIGBG * SIGL
SIGY = DSQRT( EPY * BETY(2) )
SIGDP = SIGE/PC
SIGPSQ = SIGDP * SIGDP
DPP = SQRT6*SIGDP
BETGAM = PC/RME
EPXN = EPX * BETGAM
EPYN = EPY * BETGAM

```

C STORE EPX,EPY,DP/P,EPLN

```

CALL STDAT(M,5,1,4,EM)
562 IF (FLAG) GO TO 53
563 CONTINUE

```

CALL DASH

```

WRITE (3,2) PC,GAMT,R,U0
WRITE (3,4) FREQ,T0,V0,TAUX
WRITE (3,6) RHO,TAUY,ALPHA,TAUE
WRITE (3,3) D,SIGX,PHISD,TAUQ
WRITE (3,5) TE,SIGPHI,EPSM,SIGL
WRITE (3,7) XI,DPP,SIGDP
WRITE (3,11) SIGE,SIGBG
WRITE(3,10) FK,BETGAM
WRITE (3,9) I1,I2,I3,I4,I41,I43,I5,I51,I53
WRITE (3,12) JX,JZ,JE,J1,J3,J6
WRITE (3,8) EPXN,EPYN,EPLN,EPX,EPY,EP1,EP3
CALL DASH

```

C.....

C

FORMATS

- 1 FORMAT(' PSI =', E13.5, ' —TAUQ IS .GT. VALUE PRINTED',
1 ' BELOW, WHICH CORRESPONDS TO PSI = ', E13.5)
- 2 FORMAT (' MOMENTUM',22X,' P=',E12.5,' GEV/C',10X,' TRANSITION GA
1MMA',10X,'GAMT=',E12.5,/, ' AVERAGE RADIUS',17X,' R=',E12.5,
2 ' METERS',9X,' SYNCHROTRON RADIATION',7X,'U0=',E12.5,' GEV/TURN')
- 3 FORMAT (' DAMPING PARTITION'16X,'D=',E12.5,16X,
1 ' SIGMAX(0—COUPLING)',8X,'SIGX=',E12.5,' METERS',
A/, ' PHASE ANGLE',19X,
2 'PHIS=',E12.5,' DEGREES',8X,' QUANTUM LIFETIME',10X,'TAUQ=',E12.5,
3 ' SECONDS')
- 4 FORMAT (' R F FREQUENCY',18X,' F=',E12.5,' HZ',13X,' CIRCULATION
1PERIOD',10X,'T0=',E12.5,' SECONDS',/, ' R F PEAK VOLTS',18X,'V0=',
2E12.5,' GV',14X,'HORIZONTAL DAMPING',8X,'TAUX=',E12.5,' SECONDS')
- 5 FORMAT (' ENERGY OSCILLATION PERIOD',5X,' TE=',E12.5,' SECONDS',
18X,' BUNCH LENGTH',12X,'SIGPHI=',E12.5,' RADIANS',/,
2 ' ENERGY APERTURE',15X,'EPSM=',E12.5,' GEV ',10X,
3 ' BUNCH LENGTH',14X,'SIGL=',E12.5,' METERS')
- 6 FORMAT (' EFFECTIVE RADIUS',17X,'R=',E12.5,' METERS',9X,' VERTICAL
1 DAMPING',10X,'TAUY=',E12.5,' SECONDS',/, ' MOMENTUM COMPACTION',
210X,'ALPHA=',E12.5,17X,'ENERGY DAMPING',12X,'TAUE=',E12.5,
3 ' SECONDS')
- 7 FORMAT(' (EPSM',2H**,'2/SIGE',2H**,'2)/2',11X,' XI=',E12.5,17X,
1 'MOMENTUM SPRD',1H', 'SQRT(6)',5X,'DP/P=',E12.5,/,
2 ' SIGMA(DP/P)',18X,'SIGDP=',E12.5)
- 9 FORMAT(/' I1=',E12.5,/, ' I2=',E12.5,/, ' I3=',E12.5,/, ' I4=',E12.5,
1 14X,' I41 = ',E12.5,14X,' I43 = ',E12.5,
2 /, ' I5=',E12.5,14X,' I51 = ',E12.5,14X,' I53 = 'E12.5)
- 11 FORMAT (32X,'SIGE=',E12.5,42X,'SIGBG=',E12.5)
- 10 FORMAT (/5X,'K(HARMONIC NO.) = ',E12.5,20X,'BETAGAMMA = ',E12.5)
- 8 FORMAT (///10X,'EMITTANCES (METER RADIANS)'/10X,'EPXN = ',E12.5,
1 7X,'EPYN = ',E12.5,7X,'EPLN = ',E12.5/10X,' EPX = ',E12.5,7X,
2 ' EPY = ',E12.5/11X,'EP1 = ',E12.5,7X,' EP3 = ',E12.5/)
- 12 FORMAT (/ ' JX = ',F12.5,14X,' JY = ',F12.5,14X,' JE = ',F12.5/,
1 ' J1 = ',F12.5,14X,' J3 = ',F12.5,14X,' J6 = ',F12.5)
- 13 FORMAT (1X,'CYEM'/)
- 14 FORMAT(2X'POS'11X'A11'9X'A12'9X'A22'9X'A13'9X'A14'9X'A23'9X'A24'
1 9X'A33'9X'A34'9X'A44'/
2 16X'B11'9X'B12'9X'B22'9X'B13'9X'B14'9X'B23'9X'B24'9X'B33'9X'B34'
3 9X'B44'/)
- 15 FORMAT(/1X,I4,1X,A5,1X,10F12.7/ 12X,10F12.7)

```

16  FORMAT (1X,'BEAM ELLIPSE MATRIX — SIJ = SIGMA(I,J), '
1   ' RII = SQRT(SII), TIJ = ATAN( 2.SIJ / (SII-SJJ) ) / 2 ' /
2   2X,'POS',11X,'R11',9X,'R22',9X,'T13',
3   9X,'T14',9X,'T23',9X,'T24',9X,'R33',9X,'T34',9X,'R44' /)
18  FORMAT(' BEAM ELLIPSE MATRIX — SIJ = SIGMA(I,J) ' /
1   ' POS', 11X, 'S11', 9X, 'S12', 9X, 'S22', 9X, 'S13',
2   9X,'S14',9X,'S23',9X,'S24',9X,'S33',9X,'S34',9X,'S44' /)
17  FORMAT (1X,I4,1X,A5,1X,10E12.4)
19  FORMAT(' (BETATRON PART OF SIGMA MATRIX) ')
20  FORMAT(' (TOTAL SIGMA MATRIX) ')

```

```

RETURN
END

```

```

C ***** CYX *****
      SUBROUTINE CYX(PROD)
      IMPLICIT REAL*8(A-H,O-Z)

```

```

C THIS ROUTINE...

```

```

C 1. COMPUTES K CYCLED PRODUCTS OF A MATRIX PRODUCT, PROD, WHOSE
C    FACTORS ARE STORED IN THE ARRAY, MI.
C 2. COMPUTES AND PRINTS OUT THE BETATRON FUNCTIONS OF THE CYCLED
C    PRODUCTS.
C   ITP -- WRITE TAPE5 OPTION IF GT 0.

```

```

C IF THE ORIGINAL INSTRUCTION WAS A CYB, THEN OPTIONAL OUTPUT MAY BE
C OBTAINED ON TAPE A5.

```

```

C IF KA NEGATIVE, ONLY PRINT POSITIONS OF ELEMENTS WHOSE FIRST
C CHARACTER IS ' (4-8 PUNCH)

```

```

C FOR CYA,CYC WRITE TAPE 5 OPTIONS CONTROLLED BY KA AS FOLLOWS —
C ABS(KA) = 11,12 — BETA FUNCTIONS IN BINARY, AGS SEQUENCE
C — NPOS,POS,NAME,L,LP,LPP,THETA,KV,KVP,KV2,BV,BH,AV,AH,QV,QH,X,XP
C ABS(KA) = 13,14 — XEQ,YEQ IN BCD
C ABS(KA)=12,14,16 — SUPPRESS PRINTING
C ABS(KA)=15,16 — POLARIZATION PARAMETERS ON TAPE 11
C AB(KA)=98 WRITE TAPE 98 FOR PRINTER PLOTS
C FOR BEST PLOT PARAMETERS (BX, XEQ, BY, YEQ) ON TAPE 12
C FOR FXPT BX, XCO, BY, YCO ON 12 AND POLARIZATION PARAMETERS ON 11

```

```

C IF KB.LT.0, A MATRIX CORRESPONDING TO THE REFLECTION OF THE INPUT
C BEAM LINE IS ADDED TO THAT BEAM LINE. THUS ONLY HALF OF A SYMMETRIC
C SUPERPERIOD NEED BE SPECIFIED. TUNES AND CHROMATICITIES ARE
C CALCULATED FOR THE COMPLETE SUPERPERIOD.

```

```

C **NOTE** ROUTINE USES ALL 7 SPECIAL 3X3 LOCATIONS, MEND THRU MEND-6.
C AND M7END-5.

```

```

C***** NOTES ON FILES *****

```

```

C NAMED FILES ARE IN COMMON BLOCK NAMED FILES. THEIR VALUES ARE
C SET IN BLOCK DATA.
C
C FILE 12 PLOT FILE
C FILE ORB12 MADE FOR ORBIT

```


C FILE CYBO IS VESTIGIAL. WAS MADE AS OUTPUT FILE FROM CYB COMMAND.
 C FILE JBIS IS MADE FOR IBS.
 C FILE 98 FOR PAPER PLOTS
 C FILE 11

INTEGER OP,BDAT,ELNUM

INCLUDE 'TSW.CCC'

LOGICAL PFLAG

INCLUDE 'BSTORE.CCC'

INCLUDE 'BINFF.CCC'

INCLUDE 'BWORK.CCC'

INCLUDE 'XXYY.CCC'

INCLUDE 'BPLTCOM.CCC'

INCLUDE 'CONTRL.CCC'

INCLUDE 'BMI1.CCC'

INCLUDE 'FILES.CCC'

INCLUDE 'SVNAM.CCC'

C

DIMENSION MI(16000)

EQUIVALENCE (MI,MI1),(NELS,NELS1)

INCLUDE 'MG.CCC'

INCLUDE 'MATCH3.CCC'

common/seed/nr

DIMENSION RX(2,3),RY(2,3),BETX(9),BETY(9),ML(3),GL(18)

DIMENSION PRM(6),PAR(4),V0(6)

DIMENSION TAB(10),T(49),RW(3),VW(7),VP(7)

DIMENSION MBX(2),MBY(2),MXQ(2),MYQ(2)

DIMENSION MCXX(2),MCYY(2)

DIMENSION BXP(6),BYP(6)

DIMENSION ABX(2,2),ABXI(2,2),B(2),S(2),ISFD(2),XI(2)

DIMENSION AS(1000),BS(1000),PSX(1000),PSY(1000),UU(3),LS(3)

DIMENSION MSFD(2),SVAL(2)

LOGICAL CHFLG, AMPFLG

EQUIVALENCE (GL,TAB)

EQUIVALENCE (IX1,XI1),(IX2,XI2)

INCLUDE 'DIM.CCC'

INCLUDE 'SWTCH.CCC'

INCLUDE 'KINET.CCC'

COMPLEX*16 TGAM,TGAM2

COMPLEX*16 II,ZG,ZCO,ZP

REAL*8 KV

INTEGER PROD,PR,HMAGV

LOGICAL MVF

LOGICAL PLOT,PRINT,FLG,BTABL,SKIP

LOGICAL PRPLOT

EQUIVALENCE (GL(4),XNUX),(GL(5),YNUY),(GL(8),CHRX),

1 (GL(9),CHRY),(GL(10),ALPHA),(GL(1),BXXM),

```

2      (GL(2),BYMX),(GL(3),XQMX),(GL(11),YQMX),
3      (GL(12),BXMN),(GL(13),BYMN),(GL(14),XQMN),
6      (GL(15),YQMN),(GL(16),CIRC),(GL(17),THETH),
5      (GL(18),THETY),(GL(7),TTGAM)

```

```

      INTEGER HNPOL,HZEQ,HTEQ,HMAX,HMIN
      DIMENSION HMAX(2),HMIN(2),LABEL(2)
      DATA HNPOL,HZEQ,HTEQ/4HNPOL,3HZEQ,3HTEQ/
      DATA HMAX,HMIN/4HMAXI,2HMA,4HMINI,2HMA/
      DATA CC/2.997925D8/
      DATA IV/2/

```

```

      DATA IBLNK/1H /
      DATA ZE/0./
      DATA HMAGV/4HMAGV/
      DATA II/(0.,1.D0)/
      ITER = 0

```

```

C  SAVE NO PRINT OPTION
      ISVPR = NOPR

```

```

5  PI2=6.D0*DATAN(1.D0)
      RADEG=PI2/360.D0
      FLG = .TRUE.
      SKIP = .FALSE.
      ABX(1,1)=0.
      ABX(1,2)=0.
      ABX(2,1)=0.
      ABX(2,2)=0.
      CHFLG=.FALSE.
      IF (NOPR.AND..NOT.GLOBAL) SKIP = .TRUE.
      IF (ITER.EQ.1) THEN
          NOPR = .TRUE.
          CHFLG = .FALSE.
          AMPFLG = .TRUE.
      END IF
      PRINT=.TRUE.
      PFLAG = .FALSE.
      MYF = .FALSE.
      BTABL = .FALSE.
      IF (NELS.GT.44) PFLAG = .TRUE.
      KS=0
      ITP=0
      ITAPE = 0
      NRN=nr
      LNUM=0

```

```

      IF(ERROR) RETURN
C  USES WORKING STORAGE MEND-1
      K = INFF(4,PROD)
      IF (K.GE.0) GO TO 1
      K=-K
      FLG = .FALSE.
1  KA=K
      PRPLOT = .FALSE.
      IF (KA.EQ.98) PRPLOT=.TRUE.
      KB = INFF(5,PROD)

```

```

      OP = INFF(1,PROD)
      IF (OP.NE.3HCYB) K=NELS
      IPP = INFF(6,PROD)
C   ZERO TAB
      DO 27 IC=1,10
27   TAB(IC) = 0.
      CHX=0.
      CHY=0.
      IA=0
      EL=0.
      THETH = 0.
      THETV = 0.
      KK=K
      ISK=0
      NK=0
      ETAST = 0.
C   FKK = KK
      PR = PROD
      NGO = 1
      DXCO=0.
      DYCO=0.
      YCO=0.
      XCO = 0.
      NMVAR = HZEQ
      IF (KNFLAG) NMVAR = HTEQ
      IF (OP.EQ.3HCYB) GO TO 32
      IF(OP.EQ.4HFXPT) GO TO 33
      CALL DATA(PR,2,3,2,ISFD)
C   SET CHFLG TRUE FOR SEXTUPOLE CHROM CORRECTION
      IF (ITER.GT.0) GO TO 288
      AMPFLG = .FALSE.
      NAMPL = BDAT(PROD,2)
      IAMPL = 0
      IF (NAMPL.EQ.'AMPL') IAMPL = 1
      IF (ISFD(1).NE.IBLNK.OR.ISFD(2).NE.IBLNK) CHFLG=.TRUE.
      IF (CHFLG) CALL DATA(PR,1,1,2,XI)
      IF (.NOT.CHFLG.AND.IAMPL.EQ.1) AMPFLG = .TRUE.
288  IF (OP.NE.4HCYA ) GO TO 29

      CYA
      IA=1
      NK = PR - K - 1
29   MN = MNAME(PROD)
30   IF (KA.GE.10) ITP=KA-10
      IR = PROD
      CALL MMMM(PR,KK,MI)
      KWP = KK
      IF (KB.GE.0) GO TO 301
      IF (OP.EQ.3HCYA) NK=NK-1
      KK = KK+1
      MI(KK)=MEND-6
      CALL REF(MEND-6,PR)
      ML(1)=PR
      ML(2)=MEND-6
      CALL MMMM(PR,2,ML)

```

```

301  CONTINUE
      IF (OP.EQ.3HCYA) PR = PROD - 1
      IF (OP.EQ.3HCYC.OR.OP.EQ.3HIBS) GO TO 26
      IF (OP.EQ.4HBEST) GO TO 28
310  CALL BET(PR,BETX,BETY,1)
      VP(1) = BETX(5)
      VP(2) = BETX(6)
      VP(3) = BETY(5)
      VP(4) = BETY(6)
      VP(5) = 0.
      VP(6) = 1.D0
      VP(7) = 0.
31   KW=KK+1
      MI(KW)=MI(1)
      IF(ITP.EQ.2.OR.ITP.EQ.4.OR.ITP.EQ.6) PRINT=.FALSE.
*           SET UP PLOT OPTION
      PLOT = .FALSE.
      IF (OP.EQ.4HBEST.OR.(OP.EQ.4HFXPT.AND.ITAPE.EQ.1)) GO TO 399
      GO TO 39
399  CONTINUE
      IF (MN.EQ.IBLNK) GO TO 39
      PLOT = .TRUE.
      WRITE(12,2001) MN,KW,IV
      WRITE (12,2005) BUF,BUFF
      WRITE (12,2007) NAMRUN,NRUNCOM
      GO TO 39
*           CYB
32   ITP=IDAT(PROD,1)
      MN=BDAT(PROD,1)
      PR = ELNUM(MN)
      IR = PR
      GO TO 310
C           CYC
26   IF (KA.NE.21) GO TO 28
      LQ3 = INFF(24,PR)
      LINE = LQ3
      BTABL = .TRUE.
C           BEST
28   CALL RXY(PR,RX,RY,RW)
      PR=MEND-1
      CALL STXY(PR,RX,RY,RW)
C  INITIALIZE FOR PLOT TAPE AT FIRST POSITION
      NAMOP = IBLNK
      MATI = IBLNK
      GO TO 310
*           FXPT
33   ISK = 1
      NGO = 2
      KAF=INFF(4,PR)
      IF (KAF.LT.0) FLG = .FALSE.
      KAF = JIABS(KAF)
      CALL DATA(PR,7,1,7,VW)
      IRAYS = IDAT(PR,4)
      ITAPE=IDAT(PR,5)
      FAC = 1.

```

```

IF (IRAYS.EQ.0.OR.IRAYS.EQ.2) FAC = 1000.D0
X0 = VW(1)
XP0 = VW(2)
Y0 = W(3)
YP0 = VW(4)
DPP = VW(6)
CALL DATA(PR,8,1,7,VP)
EX0 = VP(1)
EXP0 = VP(2)
EY0 = VP(3)
EYP0 = VP(4)
MN = MNAME(PROD)
ITP = 0
IF( (KA.LE.1).OR.(KA.EQ.3) ) PRINT=.FALSE.
IR = PROD
GO TO 31
39 PSIX = 0.
PSIY = 0.
DELX = 0.
DELY = 0.
I = 1
SLENI=0.
THETI=0.
CALL RXY(PR,RX,RY,RW)
*          COMPUTE TRACE
TRX = RX(1,1) + RX(2,2)
TRY = RY(1,1) + RY(2,2)
35 CONTINUE
IF ((DABS(TRX).GE.2.).OR.(DABS(TRY).GE.2.)) GO TO 125
IF (IPP.GT.0) GO TO 75
IF ((ISK.EQ.0).AND.(MODE.EQ.1).AND..NOT.NOPR) CALL DASH
GO TO (40,41) NGO
41 IF (.NOT.PRINT) GO TO 75
C WRITE FXPT HEADING
NAMFX = INFF(2,PROD)
IF (PFLAG) WRITE(3,1035) NAMFX
IF (.NOT.PFLAG) WRITE (3,1029) NAMFX
WRITE(3,1020)
IF (IRAYS.NE.0.AND.IRAYS.NE.2) GO TO 75
WRITE(3,1022)
GO TO 75
C WRITE CYX HEADING
40 IF (PFLAG) WRITE(3,1035) MN
IF (.NOT.NOPR) WRITE(3,1000) NMVAR
GO TO 75
*****

*          ITERATES FROM STATEMENTS 50 TO 90
50 NI=MI(1)
NISAVE=NI
MKN = INFF(20,NI)
GO TO (505,506) NGO
506 NF = M7END - 5
XCO = VW(1)*FAC
DXCO = VW(2)*FAC

```

```

YCO = VW(3)*FAC
DYCO = VW(4)*FAC
EX=VP(1)
EXP=VP(2)
EY=VP(3)
EYP=VP(4)
IF(I.GT.1) nr=NRR
CALL REVMAT(NF,NI,VW)
nrr=nr
NI = NF

```

C CORRECT CALCULATION OF TRANSITION GAMMA

```

505 IF (I.EQ.KW) GO TO 5051
CALL RTRV7(NI,T,RW)
CALL MXV7(T,VP,VP)
EST = -VP(5)
IF (KNFLAG) EST = EST - (EL/(GAM*GAM))
5051 CALL RXY(NI,RX,RY,RW)
IF ( ( KB .GE.0 ).OR.(I.LT.KW) ) GO TO 507
PSIX = 2.00*PSIX
PSIY = 2.00*PSIY
IPOS = I - 1
MATI = 4HREFL
GO TO (52,508) NGO
508 XCO = X0*FAC
DXCO = -XP0*FAC
YCO = Y0*FAC
DYCO = -YP0*FAC
EX=EX0
EXP=-EXP0
EY=EY0
EYP=-EYP0
GO TO 52
507 CONTINUE
PSIX = PSIX + DELX/PI2
PSIY = PSIY + DELY/PI2
DENX = BETX(2)*RX(1,1)-BETX(3)*RX(1,2)
IF (DENX.EQ.0) GO TO 502
DELX = DATAN2(RX(1,2),DENX)
IF (DELX.LT.0..AND.RW(1).GE.0) DELX = DELX + PI2

502 CONTINUE
DENY = BETY(2)*RY(1,1)-BETY(3)*RY(1,2)
IF (DENY.EQ.0) GO TO 504
DELY = DATAN2(RY(1,2),DENY)
IF (DELY.LT.0..AND.RW(1).GE.0) DELY = DELY + PI2

504 CONTINUE
IPOS = I-1
IF (IPOS.NE.0) GO TO 51
MATI = IBLNK
BXXM = BETX(2)
BYMX = BETY(2)
XQMX = BETX(5)
YQMX = BETY(5)
BXMN = BETX(2)

```

```

      BYMN = BETY(2)
      XQMN = BETX(5)
      YQMN = BETY(5)
      IF(NGO.EQ.1) GO TO 511
      XCMX=XCO
      XCMN=XCO
      YCMX=YCO
      YCMN=YCO
511  CONTINUE
      DO 510 LI = 1,2
      MCXX(LI)=I
      MCYY(LI)=I
      MBX(LI) = I
      MBY(LI) = I
      MYQ(LI) = I
510  MXQ(LI) = I
      GO TO 52
51  MK = MI(IPOS)
      MATI = INFF(2,MK)
      NAMOP = INFF(1,MK)
      GG = FDAT(MK,2)
      IF (NAMOP.EQ.3HDRF) GG = 0.
      DECODE (4,20,MATI) MATJ
20  FORMAT (A1,3X)
      IF (NGO.EQ.1) GO TO 52
      BETX(5)=EX
      BETY(5)=EY
52  IF (IPOS.EQ.0) GO TO 522
      IF (.NOT.CHFLG) GO TO 522
C    CALCULATE SUMS FOR CHROM CORRECTION
      IF (MATI.NE.ISFD(1)) GO TO 521
      ABX(1,1) = ABX(1,1) + BETX(2)*BETX(5)
      ABX(2,1) = ABX(2,1) + BETY(2)*BETX(5)
521  IF (MATI.NE.ISFD(2)) GO TO 522
      ABX(1,2) = ABX(1,2) + BETX(2)*BETX(5)
      ABX(2,2) = ABX(2,2) + BETY(2)*BETX(5)
522  IF (.NOT.PRINT) GO TO 53
      IF (IPP.GT.0) GO TO 53

.....
*    PRINT BETATRON FUNCTIONS
      IF (.NOT.FLG.AND.MATJ.NE.1H ) GO TO 53
      IF (.NOT.PFLAG) GO TO (571,58) NGO
      IF (JMOD(LNUM,45).NE.0.OR.LNUM.EQ.0) GO TO (571,58) NGO
C  NEW PAGE.  WRITE HEADING.
      GO TO (57,59) NGO
57  IF (.NOT.NOPR) WRITE(3,1040) NMVAR
571  CONTINUE
      IF (.NOT.BTABL) GO TO 573
      STORE(LINE) = PSIX
      STORE(LINE+6) = PSIY
      STORE(LINE+12) = EL
      STORE(LINE+13) = THETH
      LST=LINE
      DO 572 ICY=2,6

```

```

LST = LST+1
STORE(LST) = BETX(ICY)
572 STORE(LST+6) = BETY(ICY)
    LINE = LINE + 14
573 IF (NOPR) GO TO 574
    WRITE (3,1001) IPOS,MATI,EL,PSIX,PSIY,BETX(2),BETY(2),
1    BETX(5),BETY(5),ETAST,BETX(3),BETY(3),BETX(6),BETY(6)
    LNUM=LNUM+1
    IF (JMOD(LNUM,5).EQ.0.AND.JMOD(LNUM,45).NE.0) WRITE (3,1006)
574 ETAST = EST
    GO TO 53
59    IF (IPOS.EQ.0) GO TO 58
    IF (LNUM.NE.45) GO TO 591
C SKIP PAGE RESTORE THIS TIME
C UNIQUE PAGING WITH FIRST PAGE OF FXPT.
C EXTRA LINE AT TOP OF 1ST PAGE CAUSES AUTOMATIC PAGE RESTORE HERE.
    WRITE (3,1020)
    GO TO 592
591 WRITE (3,1030)
    WRITE (3,1020)
592 IF (IRAYS.NE.0.AND.IRAYS.NE.2) GO TO 58
    WRITE (3,1022)
58 WRITE(3,1021) IPOS,MATI,EL,PSIX,PSIY,BETX(2),BETY(2),
1    BETX(3),BETY(3),EX,EXP,EY,EYP,XCO,DXCO,YCO,DYCO
    LNUM=LNUM+1
    IF (JMOD(LNUM,5).EQ.0.AND.JMOD(LNUM,45).NE.0) WRITE (3,1006)
53 CONTINUE
    IF (PRPLOT) WRITE (98,2000) IPOS,EL,PSIX,PSIY,BETX(2),
1    BETY(2),BETX(5),BETY(5)
    if (itape.ge.1)
x WRITE(orb5) KK,IPOS,MATI,SSLEN,ZE,ZE,THETI,KV,DKV,ZE,
1    BETY(2),BETX(2),BETY(3),BETX(3),PSIY,PSIX,A,AP
2    ,AV,AVP
    IF(ITAPE.GE.1) WRITE(ORB12) EX,EXP,EY,EYP,XCO,DXCO,YCO,DYCO
    IF (IPOS.EQ.0) GO TO 54
    IF (SKIP) GO TO 54
    CALL MAXMIN (BXXM,BXMN,BETX(2),MBX,IPOS)
    CALL MAXMIN (BYMX,BYMN,BETY(2),MBY,IPOS)
    CALL MAXMIN(XQMX,XQMN,BETX(5),MXQ,IPOS)
    CALL MAXMIN(YQMX,YQMN,BETY(5),MYQ,IPOS)
    IF(NGO.EQ.1) GO TO 54
    CALL MAXMIN(XCMX,XCMN,XCO,MCXX,IPOS)
    CALL MAXMIN(YCMX,YCMN,YCO,MCYY,IPOS)
    IF (.NOT.PLOT) GO TO 54
    XX(1)=EL
    Y1(1)=BETX(2)
    Y2(1)=XCO
    Y3(1)=BETY(2)
    Y4(1)=YCO
54 ELL = EL
    IF(NGO.EQ.2) GO TO 622

* STORE FOR PLOT FILE ON OPTION
    IF (.NOT.PLOT) GO TO 56
    IF (IPOS.EQ.0) GG = 0.

```



```

XX(I) = EL
Y1(I) = BETX(2)
Y2(I) = BETX(5)
Y3(I) = BETY(2)
Y4(I) = BETY(5)
WRITE (12,2050) NAMOP,MATI,GG,XX(I),Y1(I),Y2(I),Y3(I),Y4(I)
GO TO 55
56 CONTINUE
IF (ITP.EQ.0) GO TO 55
IF (ITP.GT.2) GO TO 62
KV=0.
IF (NAM.EQ.3HMAG) KV=-GK
DKV=0.
IF (NAM.EQ.4HSXTP) DKV=-GK2
IF (NGO.EQ.2) GO TO 63
A = BETX(5)
AP = BETX(6)
AV = BETY(5)
AVP = BETY(6)
GO TO 64
63 A = VW(1)/DPP
AP = VW(2)/DPP
64 CONTINUE
SSLEN=SLENI
IF (OP.EQ.3HIBS) SSLEN=EL
WRITE(JBIS) KK,IPOS,MATI,SSLEN,ZE,ZE,THETI,KV,DKV,ZE,
1 BETY(2),BETX(2),BETY(3),BETX(3),PSIY,PSIX,A,AP
2 ,AV,AVP
IF (.NOT.FLG.AND.MATJ.NE.1H ) GO TO 55
IF (OP.EQ.3HIBS) GO TO 55
WRITE(3,2)KK,IPOS,MATI,SLENI,ZE,ZE,THETI,KV,DKV,ZE,
1 BETY(2),BETX(2),BETY(3),BETX(3),PSIY,PSIX,A,AP
2 ,AV,AVP
2 FORMAT (1X,2I4,1X,A5,1X,17F6.2)
GO TO 55
62 IF (ITP.GE.5) GO TO 622
IF (I.EQ.KW) GO TO 55
WRITE (CYBO,1002) MN,IPOS,KK,EL,BETX(5),BETY(5),BETX(7),
X BETY(7)
GO TO 55
C WRITE POLARIZATION PARAMETERS ON TAPE 11
622 RHOI = 0.
GK=0.
XI1=0.
XI2=0.
IF (IPOS.EQ.0) GO TO 3333
NAM=INFF(1,MK)
IF (NAM.EQ.4HMOVE) GO TO 3335
IF (NAM.EQ.3HMAG) GO TO 3336
GO TO 3333
3335 MMK=MDAT(MK,1)
IF (INFF(1,MMK).NE.4HMAG ) GO TO 3333
MK=MMK
3336 CONTINUE
CALL DATA(MK,1,1,6,PRM)

```

```

RHOI=PRM(4)/PRM(3)
GK=-PRM(2)/PRM(3)
XI1=PRM(5)
XI2=PRM(6)
IF(IX1.EQ.1H$.OR.IX2.EQ.1H$) GO TO 3334
XI1=XI1*RADEG
XI2=XI2*RADEG
GO TO 3333
3334 XI1=PRM(1)*RHOI/2.D0
    XI2=XI1
3333 CONTINUE
    IF(I.EQ.1) WRITE(11) KW
    ZZ=0.
    IF (NGO.EQ.2) ZZ = YCO/FAC
    WRITE (11) ELL,PSIY,BETY(2),RHOI,GK,XI1,XI2,ZZ
*          COMPUTE LENGTH
55  EL = EL + RW(1)
60  I = I + 1
    IF (I.GT.KW) GO TO 90
    SLENI = RW(1)
    THETI = RW(2)
    NAM = INFF(1,NI)
    IF ((I.EQ.KW).AND.(KB.LT.0)) GO TO 65
    IF (NAM.NE.HMAGV) THETH = THETH + RW(2)
    IF (NAM.EQ.HMAGV) THETV = THETV + RW(2)
    GO TO 66
65  THETH = THETH + THETH
    THETV = 2.D0*THETV
66  CONTINUE

C  COMPUTE BETATRON FUNCTIONS FOR THE I-TH CYCLED MATRIX
C
C  IN CYA CASE, DO SIMILARITY TRANSFORMATION AND CALL BET
C  IN OTHER CASES, COMPUTE NEW BETA FUNCTIONS DIRECTLY
C
    IF (OP.NE.3HCYA) GO TO 7503
    ML(1) = -NI
    ML(2) = PR + IA
    ML(3) = NI
    IF (PR.EQ.NK) PR=MEND-1
    CALL MAM (PR,3,ML)
7503 CONTINUE
    DO 555 JJ=1,6
    BXP(JJ)=BETX(JJ)
555  BYP(JJ)=BETY(JJ)
    AXS=BETX(3)
    AYS=BETY(3)
    SS=RW(1)
75  IF (I.EQ.1.OR.OP.EQ.3HCYA) CALL BET(PR,BETX,BETY,1)

C  COMPUTE CHROMATICITY IF ELEMENT IS MAGNET
    IF(I.EQ.1) GO TO 775
    IF (SKIP) GO TO 775
    IF (OP.EQ.3HCYA) GO TO 7501
    IF (BXP(2).EQ.0) GO TO 7502

```

```

      BETX(3) = (1.D0+2.D0*RX(1,2)*RX(2,1))*BXP(3)-RX(1,1)*RX(2,1)
      1*BXP(2)-RX(1,2)*RX(2,2)*BXP(4)
      BETX(2) = (RX(1,2)**2+(RX(1,1)*BXP(2)-RX(1,2)*BXP(3))**2)
      1 /BXP(2)
      BETX(4) = (1.D0+BETX(3)**2)/BETX(2)
      BETX(5) = RX(1,1)*BXP(5)+RX(1,2)*BXP(6)+RX(1,3)
      BETX(6) = RX(2,1)*BXP(5)+RX(2,2)*BXP(6)+RX(2,3)
7502 IF (BYP(2).EQ.0) GO TO 7501
      BETY(3) = (1.+2.*RY(1,2)*RY(2,1))*BYP(3)-RY(1,1)*RY(2,1)
      1*BYP(2)-RY(1,2)*RY(2,2)*BYP(4)
      BETY(2) = (RY(1,2)**2+(RY(1,1)*BYP(2)-RY(1,2)*BYP(3))**2)
      1/BYP(2)
      BETY(4) = (1.D0+BETY(3)**2)/BETY(2)
      BETY(5) = RY(1,1)*BYP(5)+RY(1,2)*BYP(6)+RY(1,3)
      BETY(6) = RY(2,1)*BYP(5)+RY(2,2)*BYP(6)+RY(2,3)
7501 CONTINUE
      NI = NISAVE
      NAM=INFF(1,NI)
      IF(NAM.EQ.4HSXTP) GO TO 785
760  CONTINUE
      IF (NAM.NE.4HKICK.AND.NAM.NE.4HMOVE) GO TO 765
      NKIK=MDAT(NI,1)
      NOP=INFF(1,NKIK)
      IF (NOP.NE.3HMAG.AND.NOP.NE.4HMAGV) GO TO 765
      IF (MKN.EQ.3HMAP) GO TO 775
      CALL DATA(NKIK,1,1,6,PRM)
      GK=PRM(2)/PRM(3)
      IF(NOP.EQ.4HMAGV) MVF=.TRUE.
      GO TO 770
765  IF(NAM.NE.4HMAG .AND.NAM.NE.4HMAGV) GO TO 775
      CALL DATA(NI,1,1,6,PRM)
      GK=PRM(2)/PRM(3)
      IF(NAM.EQ.4HMAGV) MVF=.TRUE.
770  CONTINUE
      NZ=NI
      IF (NAM.EQ.4HKICK.OR.NAM.EQ.4HMOVE) NZ=NKIK
      RI=PRM(4)/PRM(3)
      XI1=PRM(5)
      XI2=PRM(6)
      IF (IX1.EQ.1H$.OR.IX2.EQ.1H$) GO TO 7701
      TH1=XI1*RADEC
      TH2=XI2*RADEC
      GO TO 7702
7701 TH1=PRM(1)*RI/2.D0
      TH2=TH1
7702 TAN1=DTAN(TH1)
      TAN2=DTAN(TH2)
      IF(MVF) GO TO 7703
      CALL CHRM(GK,RI,SS,TAN1,TAN2,BETX,BETY,BXP,BYP,CRX,CRY)
      GO TO 786
7703 MVF = .FALSE.
      CALL CHRM(GK,RI,SS,TAN1,TAN2,BETY,BETX,BYP,BXP,CRY,CRX)
      GO TO 786
785  CALL DATA(NI,1,1,4,PRM)
      SXLEN=PRM(1)

```

```

GK2=PRM(2)/PRM(3)
IF (.NOT.AMPFLG) GO TO 5601
IF(NAM.NE.4HSXTP) GO TO 5601
C STORE PARAMETERS FOR NONLINEAR AMPLITUDE CALCULATION
DKV=GK2
KS=KS+1
AF=DKV*DSQRT(BETX(2))
IF(SLENI.NE.0.) AF=AF*SLENI
AS(KS)=AF*BETX(2)
BS(KS)=AF*BETY(2)
PSX(KS)=PSIX
PSY(KS)=PSIY
5601 CONTINUE
CRX = 2.D0*(BETX(3)-AXS)
CRY=2.D0*(BETY(3)-AYS)
KAS=INFF(4,NI)
KBS = INFF(5,NI)
IF (KAS.EQ.0) KBS= 0
IF (KAS.FQ.0) KAS=3
MP=KAS-3
IF(MP.LT.0) GO TO 786
ZCO = ((XCO + SXLEN*DXCO/2.D0) + II*(YCO + SXLEN*DYCO/2.D0))
1 /1000.D0
ZP= BETX(5)-SXLEN*BETX(6)/2.D0 + II*(BETY(5)-SXLEN
1 *BETY(6)/2.D0)
BXC = BETX(2) + SXLEN*(BETX(3)+BETX(4)*SXLEN/4.D0)
BYC = BETY(2) + SXLEN*(BETY(3)+BETY(4)*SXLEN/4.D0)
ZG=-2.D0*ZP*GK2
IF (KBS.EQ.1) ZG=II*ZG
IF(SXLEN.NE.0.) ZG=ZG*SXLEN
IF(MP.EQ.0) GO TO 7852
DO 7851 MM=1,MP
7851 ZG=ZG*ZCO/MM
7852 CRX= CRX + DREAL(ZG)*BXC
CRY = CRY - DREAL(ZG)*BYC
786 CHX = CHX + CRX
CHY = CHY + CRY
775 CONTINUE
C CHANGE INDEX TO SAVE CYCLED MATRICES IF CYA INSTRUCTION.
IF (OP.NE.3HCYA.OR.1.EQ.KW) GO TO 50
PR=PR-1
GO TO 50

```

COMPLETES LOOP FROM STATEMENTS 50 TO 90

```

90 CONTINUE
nr=nrn
IF (SKIP) GO TO 201
BXX = BXX
BYX = BYX
91 CONTINUE
TKB = JIABS(KB)
IF(KB.EQ.0) TKB=1.D0
XNUX=TKB*PSIX
YNUY=TKB*PSIY

```

```

CALL RXY(IR,RX,RY,RW)
THETH = THETH * TKB
THETV=THETV*TKB
RLEN=ELL/PI2
RLEN = RLEN * TKB
CIRC = ELL*TKB
STOTI = 1.D0/ELL
ETASTOT = ETAST*STOTI
ALPHA = -VP(5)*STOTI
C IPP.GT.0 PRINTS ONLY TABULAR VALUES.
  IF (IPP.GT.0) GO TO 100
  IF (.NOT.NOPR) WRITE (3,1006)
  CHM =-TKB*0.25D0/PI2
  IF (KB.LT.0) CHM = 2.D0*CHM
  CHRX = CHM*CHX
  CHRY = CHM*CHY
  IF (.NOT.CHFLG) GO TO 100
C      CALCULATE CHROM CORRECTIONS
  ABINV = 1.D0/(ABX(1,1)*ABX(2,2) - ABX(1,2)*ABX(2,1))
  ABXI(1,1) = ABX(2,2)*ABINV
  ABXI(2,1) = -ABX(2,1)*ABINV
  ABXI(1,2) = -ABX(1,2)*ABINV
  ABXI(2,2) = ABX(1,1)*ABINV
C
  SPN = JIABS(KB)
  IF (SPN.EQ.0) SPN = 1.D0
  IF (KB.LT.0) SPN = SPN*2.D0
  CON = 2.D0*PI2/SPN
C
  B(1) = CON*(XI(1) - CHRX)
  B(2) = CON*(-XI(2) + CHRY)
C
  DO 99 I=1,2
    S(I) = 0.
  DO 99 J=1,2
99  S(I) = S(I) + ABXI(I,J)*B(J)
C
C      APPLY CHROM CORRECTIONS TO 2ND PARAMETER OF SEXTUPOLES
  DO 95 I=1,2
    MSFD(I) = ELNUM(ISFD(I))
    SVAL(I) = FDAT(MSFD(I),2)
    SVAL(I) = SVAL(I) + S(I)
    STORE(LOCC) = SVAL(I)
95  CONTINUE
100 IF (BEND.EQ.0.) GO TO 101
C COMPUTE TRANSITION GAMMA
  IF (ALPHA.EQ.0.) GO TO 101
  TGAM2 = 1.D0/ALPHA
  TGAM = CDSQRT(TGAM2)
  TTGAM = DIMAG(TGAM)
  IF (TTGAM.NE.0.) TAB(6) = -TTGAM
  TTGAM = REAL(TGAM)
  IF (TTGAM.NE.0.) TAB(6) = TTGAM
  GO TO 102
101 TGAM = 0.

```

```

102 IF (IPP.GT.0) GO TO 103
   IF (NOPR) GO TO 104
   WRITE(3,1008) CIRC,THETH,XNUX,CHRX,RLEN,THETV,YNUY,CHRY
   IF (BEND.NE.0) WRITE (3,1026) ALPHA,TGAM
   IF (BEND.EQ.0) WRITE (3,1025) ALPHA
104 IF (.NOT.KNFLAG) GO TO 105
   CCBET = CC * BETT
   T0 = 0.
   IF (CCBET.NE.0.) T0 = ELL/CCBET
   ETA = ALPHA - (1.D0/(GAM*GAM))
   IF (NOPR) GO TO 201
   WRITE (3,1028) ETASTOT,T0,ETA
105 IF (NOPR) GO TO 201
   LABEL(1) = HMAX(1)
   LABEL(2) = HMAX(2)
   WRITE (3,1005)
   WRITE (3,1007) LABEL,MBX(1),BXXMX,MBY(1),BYMX,MXQ(1),XQMX,MYQ(1),
1   YQMX
   LABEL(1) = HMIN(1)
   LABEL(2) = HMIN(2)
   WRITE (3,1007) LABEL,MBX(2),BXXMN,MBY(2),BYMN,MXQ(2),XQMN,MYQ(2),
1   YQMN
   IF(NGO.EQ.2)WRITE(3,1051) MCXX(1),XCMX,MCYY(1),YCMX,MCXX(2),XCMN,
1MCYY(2),YCMN
1051 FORMAT(' MAXIMA'6X'XCO('I5,')='F10.5,5X'YCO('I5,')='F10.5/
1      ' MINIMA'6X'XCO('I5,')='F10.5,5X'YCO('I5,')='F10.5)
   IF (CHFLG) WRITE(3,1045) S(1),S(2),SVAL(1),SVAL(2)
1045 FORMAT (//2X,'SEXTUPOLE CORRECTIONS —— DKSF =' ,E15.8,4X,
1'DKSD =' ,E15.8,4X, 'KSF =' ,E15.8,4X, 'KSD =' ,E15.8)
201 IF (.NOT.AMPFLG) GO TO 210
   IF(KS.EQ.0) GO TO 210
   IF(KB.GE.0) GO TO 410
   DO 411 I=1,KS
      KT=KS+KS+1-I
      AS(KT)=AS (I)
      BS(KT)=BS(I)
      PSX(KT)=PSIX-PSX(I)
411   PSY(KT)=PSIY-PSY(I)
      KS=KS+KS
410 PI=PI2/2.
   PX=PI*PSIX
   PY=PI*PSIY
   PX3=3.*PX
   PCP=PY+PY+PX
   PCM=PY+PY-PX
   SX1=DSIN(PX)
   SX3=DSIN(PX3)
   SP=DSIN(PCP)
   SM=DSIN(PCM)
   H11=0.
   H22=0.
   H12=0
   DO 402 I=1,KS
      DO 402 J=1,KS
         PHX=PX-PI2*DABS(PSX(I)-PSX(J))

```

```

PHY=PY-PI2*DABS(PSY(I)-PSY(J))
C1=DCOS(PHX)/SX1
C3=DCOS(3.*PHX)/SX3
CP=DCOS(2.*PHY+PHX)/SP
CM=DCOS(2.*PHY-PHX)/SM
AA=AS(I)*AS(J)
AB=AS(I)*BS(J)
BB=BS(I)*BS(J)
H11=H11-AA*(C3+3.*C1)
H12=H12-BB*(CP+CM)+2.*AB*C1
402   H22=H22-BB*(CP-CM+4.*C1)
      UU(1)=TKB*H11/PI2/64.
      UU(2)=TKB*H12/PI2/32.
      UU(3)=TKB*H22/PI2/64.
      DO 420 I=1,3
        LS(I)=3H -
        IF(UU(I).GE.0.) LS(I)=3H +
420   UU(I)=ABS(UU(I))
      IF (ITER.EQ.0) WRITE (3,405)
405   FORMAT (' ')
      WRITE(3,404) XNUX,LS(1),UU(1),LS(2),UU(2),CHRX,
1     YNUY,LS(2),UU(2),LS(3),UU(3),CHRY
404   FORMAT(' AMPLITUDE DEPENDENCE OF TUNES DUE TO SEXTUPOLES'/
1     ' NU-X = 'F10.6,a3,E9.3,'EX ',a3,E9.3,'EY',
2     5X,'DNUX/(DPP/P) = 'F10.5/
3     ' NU-Y = 'F10.6,a3,E9.3,'EX ',a3,E9.3,'EY',
4     5X,'DNUY/(DPP/P) = 'F10.5)
      GO TO 210
C   PRINT TAB VALUES
103   IF (NOPR) GO TO 210
      WRITE (3,1112) TAB(1),TAB(3),TAB(4),TAB(2),TAB(5),TGAM
210   IF (ISK.EQ.0) CALL DASH
      KNFLAG = .FALSE.
      IF (.NOT.BTABL) GO TO 216
      LINE = LINE - 1
      DO 215 ICY = 1,10
215   STORE(LINE+ICY) = GL(ICY)
216   IF (NGO.EQ.2) RETURN
      CALL STDAT(PROD,7,1,10,TAB)
C   RESTORE NOPR
      IF (ITER.EQ.1) THEN
        NOPR = ISVPR
        RETURN
      ELSE
        IF (.NOT.CHFLG.OR.IAMPL.EQ.0) RETURN
      END IF
C   RECALCULATE SXTIP WITH NEW VALUES AND REPEATE CYX CALCULATION
      DO 130 IVX=1,2
        CALL DATA (MSFD(IVX),1,1,4,PAR)
        PAR(4) = 0.
        DO 131 IVV=1,6
131   V0(IVV) = 0.
130   CALL SXTIP(MSFD(IVX),V0,PAR)
      ITER = 1
      GO TO 5

```

C ERROR EXIT

```

125  CONTINUE
      ERROR = .TRUE.
      IF (IPP.GT.0) GO TO 210
      WRITE (3,1009) MN,TRX,TRY

```

• COMPUTE AND PRINT LENGTH

```

      KE = KW - 1
      EL = 0.
      DO 127 IK=1,KE
      M1 = MI(IK)
      CALL DATA(M1,5,1,1,FL)
      EL = EL + FL
127  CONTINUE
      WRITE (3,1010) EL
      GO TO 210

```

.....

• FORMATS

```

1009 FORMAT (//1H ,4X,43H ***** BETATRON FUNCTIONS UNSTABLE THROUGH ,
1  A5/5X,'X TRACE = ',E12.5,10X,'Y TRACE = ',E12.5)
1010 FORMAT (10X,'TOTAL BEAM LENGTH = ',F10.5)
1000 FORMAT (
1  3X,'POS',10X,'S(M) ',5X,'NUX',6X,'NUY',6X,'BETAX(M)',4X,
2  'BETAY(M)',4X,'XEQ(M)',4X,'YEQ(M)',2X,A4,'(M)',3X,'ALPHAX',
3  4X,'ALPHAY',5X,'DXEQ',5X,'DYEQ'/1X,131(1H-))
1001 FORMAT (1X,I5,1X,A4,F11.4,2F9.5,2F12.5,2F10.5,F9.4,2F10.5,
1  2F9.5)
1002 FORMAT(A5,2I3,5F14.8)
1005 FORMAT (2H )
1006 FORMAT (1X)
1007 FORMAT (2X,2A4,2X,'——',1X,'BETX(',I5,') = ',F12.5,4X,'BETY(',
1  I5,') = ',F12.5,5X,'XEQ(',I5,') = ',F12.5,5X,'YEQ(',I5,
2  ') = ',F12.5)
1008 FORMAT (/
1  2X,'CIRCUMFERENCE = ',F11.4,1X,'K',10X,'THETA = ',F12.8,
2  1X,'RAD',6X,'NUX = ',F10.5,6X,'DNUX/(DP/P) = ',F10.5/
3  9X,'RADIUS = 'F10.4,' M'10X'THETA = 'F12.8,
4  1X,'RAD',6X,'NUY = ',F10.5,6X,'DNUY/(DP/P) = ',F10.5)
1111 FORMAT (15X,'TGAM=(',F10.5,',',F10.5,')')
1112 FORMAT (5X,'BETATRON FUNCTIONS',7X,'MAX BETX = ',F10.5,4X,'MAX XEQ
1  = ',F10.5,4X,'NUX = ',F10.5/30X,'MAX BETY = ',F10.5,28X,'NUY = ',
2  F10.5/30X,'TRANSITION GAMMA = (',F10.5,',',F10.5,')')
1900 FORMAT (2X,'CHROMATICITIES FROM QUADRUPOLES EXPLICITLY APPEARING'
1  ' IN 'A5, ' —— DNUX/(DP/P) = ',F10.4,3X,'DNUY/(DP/P) = ',F10.4)
1020 FORMAT (1X,'POS',10X,'S',6X,'QX',4X,'QY',5X,'BX',7X,'BY',6X,
1  'AX',5X,'AY',4X,'EX',8X,'EXP',4X,'EY',5X,'EYP',9X,'XCO',6X,
1  'DXCO',5X,'YCO',6X,'DYCO')
1021 FORMAT(1X,I4,1X,A4,F8.2,2F6.2,2F9.4,2F7.2,2F9.5,5F9.4,F8.4)
1022 FORMAT (13X,'(M)',17X,'(M)',6X,'(M)',19X,'(M)',13X,'(M)',16X,
1  '(MM)',5X,'(MR)',5X,'(MM)',5X,'(MR)')
1025 FORMAT (' (DS/S)/(DP/P)= ',F10.7)
1026 FORMAT (' (DS/S)/(DP/P)= ',F10.7,
1  14X,'TGAM=(',F10.5,1H,F10.5,1H))
1028 FORMAT (2X,'(DT/T)/(DP/P) = ',F10.5,14X,'T0 = ',E12.5, 'SEC',

```



```

      1 6X,'ETA = 1/'TGAMSQ - 1/GAMSQ = ',F10.5)
1029 FORMAT (/1X,'BETATRON FUNCTIONS OF ',A5/)
1030 FORMAT (1H1)
1035 FORMAT (1H1,'BETATRON FUNCTIONS OF ',A5)
1040 FORMAT (1H1,1X,
      1 1X,'POS',10X,'S(M) ',4X,'NUX',5X,'NUY',6X,'BETAX(M)',4X,
      2 'BETAY(M)',5X,'XEQ(M)',5X,'YEQ(M)',3X,A4,'(M)',3X,'ALPHAX',
      3 4X,'ALPHAY',5X,'DXEQ',4X,'DYEQ'/1X,131(1H-))
2000 FORMAT(1X,I4,1X,7F18.10)
2001 FORMAT (1X,A5,1X,I10,1X,I1,1X,'*****')
2005 FORMAT (1X,A9,1X,A8)
2007 FORMAT (1X,A5,1X,15A4)
2050 FORMAT (1X,2A5,1P6E13.5)
      END

C ***** DASH *****
      SUBROUTINE DASH
C      WRITE ONE LINE OF DASHES
      WRITE (3,2)
      2 FORMAT (1X,131(1H-))
      RETURN
      END

C ***** DATA *****
      SUBROUTINE DATA(M,I,J,K,A)
      IMPLICIT REAL*8 (A-H,O-Z)
C      READS K DATA OF TYPE I STARTING AT INDEX J INTO ARRAY A
C
C      IF I=1, PICK UP FROM FL. PT. STORAGE
C      IF I=2, PICK UP FROM BCD, I=3 FROM INTEGER.
C      IF I=4, SKIP. I=5, PICK UP FROM LQ, I=7 FROM LQ2, I=8 FROM LQ3
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      DIMENSION ISTORE(1),LOC(10),A(1)
      INCLUDE 'FLT.N.CCC'

      EQUIVALENCE (ISTORE(1),STORE(1))
      INTEGER SHFTR,SHFTL
      INTEGER ELNUM
      LOGICAL NEG
      DIMENSION IA(2)
      EQUIVALENCE (AA,IA(1)),(A1,IA(1))
      EQUIVALENCE (FVAR,IVAR)
      INTEGER FFF0 /ZFFFFFFF00/, IdoI /1H$/
      DATA MASK/ZFF000000/
      DATA MBLK/Z20000000/

      NEG = .FALSE.
      IF (I.EQ.4) RETURN
      LOC(1)=INFF(14,M)
      LOC(2)=INFF(16,M)
      LOC(3)=INFF(12,M)
      LOC(5) = INFF(10,M)
      LOC(6)=INFF(14,M)
      LOC(7)=INFF(19,M)
      LOC(8)=INFF(24,M)

```

```

      IF((I .EQ. 2) .OR. (I .EQ. 3) ) GO TO 15
      LOCI=LOC(I)+J-2
      DO 10 L=1,K
      LOCI=LOCI+1
      A(L) = STORE(LOCI)
      IF (I.NE.1) GO TO 10
      AA = A(L)
      NR=IA(2)
      NL = LAND(FFF0, IA(1))

      IF (NL.EQ.0.AND.NR.NE.0) GO TO 100
C   FLOATING POINT
      GO TO 10
C   SYMBOLIC. LEFT ADJUST,FIND VARIABLE AND ITS VALUE
      100 IVAR=NR
C   IF doi SIGN, LEAVE AS IS.
      IF (IVAR .NE. Idoi) GO TO 24

      A(L)=FVAR
      GO TO 10
24   CONTINUE
      MSK = SHFTL(NR,24)

      IF (MSK.EQ.IFL(1)) NEG = .TRUE.
      IF (.NOT.NEG) GO TO 26
      MSK = SHFTR(NR,8)
      IVAR = MSK.OR.MBLK

26   MN = ELNUM(IVAR)
      INDEX=INFF(10,MN)
      A(L)=STORE(INDEX)
      IF(NEG) A(L)=-A(L)

      NEG = .FALSE.
10   CONTINUE
      GO TO 20
C   BCD OR INTEGER
      15   LOCI=LOC(I)*2-1
      LOCI=LOCI+J-2
      K1=(K+1)/2
      DO 16 L=1,K1
      LOCI=LOCI+1
      IA(1)=ISTORE(LOCI)
      LOCI=LOCI+1
      IA(2)=ISTORE(LOCI)
      A(L)=A1
16   CONTINUE
20   CONTINUE
      RETURN
      END

```

```

C ..... DEFSET .....
      SUBROUTINE DEFSET(M,NM,KC,MXLIST,IOP,KN)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C  RESERVE SPACE FOR KC NUMBER OF CYCLED MATRICES WHOSE NAMES
C  CONSIST OF THE FIRST LETTER OF NM, FOLLOWED BY NUMBERS IN SEQUENCE.
C  SETS UP SPACE FOR TWO MAGNETS AND OPTIONAL NUMBER OF DRIFTS.
C      IOP = TOTAL NUMBER OF MAGS AND DRFS
C      KN = KIND FOR INFO
C
      INCLUDE 'BMI.CCC'

      EQUIVALENCE (LIST(1),MI(1))
      DIMENSION LIST(1),MXLIST(1)
      INTEGER NEGZER/-32767/
      Idol=NEGZER
      IF (KC.EQ.0) GO TO 536
      M1=M-1
      CALL EQU(M1,M)
C      GENERATE NUMBERED NAMES
      CALL GENNUM(NM,KC,LIST)
      DO 535 JA=1,KC
      CALL INFW(LIST(JA),2,M1)
530  IF (JA.EQ.1) GO TO 531
      CALL RESRV8(M1,KN,Idol,0,0,0,0,0)
531  M1=M1-1
535  CONTINUE
C  IF IOP=0,RESERVE FOR CYCLED MATRICES ONLY
536  IF (IOP.EQ.0) RETURN
      M1 = M - KC
C  RESERVE FOR 2 MAGNETS
      I1=2
      IF(IOP .EQ. 1) I1=1
      DO 292 I = 1,I1
      M1=M1-1
      CALL MAGRSV(-M1,MXLIST(I),0,0,NM)
292  CONTINUE
      IF(IOP .EQ. I1) RETURN
C      RESERVE FOR DRIFTS
      I1=I1+1
      DO 293 I=I1,IOP
      M1=M1-1
      CALL DRFRSV(-M1,MXLIST(I),0,0,NM)
293  CONTINUE
      RETURN
      END
C ***** DERIV *****
      SUBROUTINE DERIV(S,Y,DY)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Y(1),DY(1)
      INCLUDE 'TRKINT.CCC'
C  ENVELOPE EQUATIONS
      DY(1) =Y(2)
      F1 = G/(Y(1)+Y(3))
      F2 = EX**2/Y(1)**3
      DY(2) = F1 + F2 - OMSQ*Y(1)
      DY(3) = Y(4)
      F2 = EY**2/Y(3)**3
      DY(4) = F1 + F2 - OMSQ*Y(3)

```

```

      RETURN
      END
C ***** DERIVE *****
      SUBROUTINE DERIVE(GG,GG2)
      IMPLICIT REAL*8 (A-H,O-Z)
CC      CALCULATES THE FIRST DERIVATIVES OF FCN (GG),
CC      EITHER BY FINITE DIFFERENCES OR BY TRANSFORMING THE USER-
CC      SUPPLIED DERIVATIVES TO INTERNAL COORDINATES,
CC      ACCORDING TO WHETHER ISW(3) IS ZERO OR ONE.
CC      IF ISW(3) = 0, AN ERROR ESTIMATE GG2 IS AVAILABLE
CC
      DIMENSION GG(30),GG2(15) , GY(30)
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /VARIAN/ V(15,15)
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      INCLUDE 'PAREXT.CCC'
      IF (ISW(3) .EQ. 1) GO TO 100
      IFLAG = 4
      DO 46 I=1,NPAR
      EPS = 0.1D0 * DABS(DIRIN(I))
      IF (ISW(2) .GE. 1) EPS = EPS + 0.005D0*DSQRT(V(I,I)*UP)
      IF (EPS .LT. 1.0D-8*DABS(X(I))) EPS = 1.0D-8*X(I)
      XTF = X(I)
      X(I) = XTF + EPS
      CALL INTOEX(X)
      CALL FCN(NPAR,GY,FS1,U ,IFLAG)
      NFCN=NFCN+1
      X(I) = XTF - EPS
      CALL INTOEX(X)
      CALL FCN(NPAR,GY,FS2,U,IFLAG)
      NFCN=NFCN+1
C      . . . . . FIRST DERIVATIVE
      GG(I)= (FS1-FS2)/(2.0D0*EPS)
C      . . . ERROR ON FIRST DERIVATIVE
      GG2(I)= (FS1+FS2-2.0D0*AMIN)/(2.0D0*EPS)
      X(I) = XTF
46 CONTINUE
      CALL INTOEX(X)
      GO TO 200
C      . DERIVATIVES CALC BY FCN
100 DO 150 I= 1, NU
      LC=LCORSP(I)
      IF (LC .LT. 1) GO TO 150
      IF (LCODE(I) .GT. 1) GO TO 120
      GG(LC)=GG(I)
      GO TO 150
120 DD = (BLIM(I)-ALIM(I))*0.5D0 *DCOS(X(LC))
      GG(LC)=GG(I)*DD
150 CONTINUE
200 RETURN
      END
C*****DELQ*****

```

```

SUBROUTINE DELQ(S,W,DW)
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'CDERIV.CCC'
EQUIVALENCE (X,T(1)),(Y,T(2)),(Z,T(3)),
1          (GX,G(1)),(GY,G(2)),(GZ,G(3)),(GV,G(4))
DIMENSION T(3),G(4)
DIMENSION W(4,1),DW(4,1)
C
C  ENVELOPES
      EPS2=EPS*EPS
      AX=W(1,1)
      AY=W(3,1)
      AX3=AX*AX*AX
      AY3=AY*AY*AY
      AXY=AX+AY
      EX=EPS2/AX3
      EY=EPS2/AY3
      QXY=Q/AXY
      FX=EX+QXY
      FY=EY+QXY
C  EL. QUAD.
      X=AX
      Y=AY
      Z=S
      CALL ELQ (LOCALPH,NUMALPH,NU,L,T,G)
C
      DW(1,1) = W(2,1)
      DW(3,1) = W(4,1)
      DW(2,1) = FX + GX
      DW(4,1) = FY + GY
C
      RETURN
      END
C***** BEAM *****
      SUBROUTINE BEAM(M)
      RETURN
      END
C***** DOFIT *****
      SUBROUTINE DOFIT(M)
      RETURN
      END
C***** SCOP *****
      SUBROUTINE SCOP(M)
      RETURN
      END
C*****STR2*****
      SUBROUTINE STR2(M)
      RETURN
      END
C*****STRPI*****
      SUBROUTINE STRPI(M)
      RETURN
      END

```

C..... TEST

```

SUBROUTINE TEST(M)
RETURN
END

```

C DER1

```

SUBROUTINE DER1(Z,W,DW)
IMPLICIT REAL*8 (A-H,O-Z)

```

C INTEGRATION THROUGH WIGGLER MAGNET

```

INCLUDE 'CDERIV.CCC'

```

```

DIMENSION W(4),DW(4),A(3),B(3)

```

```

EQUIVALENCE (L,DAT(1)),(LAMBDA,DAT(2)),(BRHO,DAT(3)),

```

```

1      (BV,DAT(4)),(A0,DAT(5))

```

```

DATA NH/3/

```

```

REAL*8 L,LAMBDA,K

```

```

IF (IFLAG) 1,1,2

```

```

1      J = 0

```

```

DO 12 I=2,6,2

```

```

J = J + 1

```

```

A(J) = DAT(5+I)

```

```

12     B(J) = DAT(6+I)

```

```

BR=BRHO*(1.+DP)

```

```

BVBR=BV/BR

```

```

K=4.*DARCOS(0.D0)/LAMBDA

```

```

DW(2) = 0.

```

```

2      X=W(1)

```

```

PX=W(2)

```

```

Y=W(3)

```

```

PY=W(4)

```

```

DX=DW(1)

```

```

DPX=DW(2)

```

```

DY=DW(3)

```

```

DPY=DW(4)

```

```

UY=0.

```

```

U=A0*Z

```

```

DO 3 N=1,NH

```

```

XK=FLOAT(N)*K

```

```

PHIY=XK*Y

```

```

PHIZ=XK*Z

```

```

AB=A(N)*DSIN(PHIZ)-B(N)*DCOS(PHIZ)

```

```

E=DEXP(PHIY)

```

```

EI=1./E

```

```

CH=(E+EI)/2.

```

```

SH=(E-EI)/2.

```

```

U=U+AB*CH/XK

```

```

3      UY=UY-AB*SH

```

```

U=U*BVBR

```

```

UY=UY*BVBR

```

```

PXM=PX-U

```

```

HAM=-DSQRT(1.-PXM*PXM-PY*PY)

```

```

DX=-PXM/HAM
DY=-PY/HAM
DPY=PXM*UY/HAM
JW(1) = DX
DW(3) = DY
DW(4) = DPY

RETURN
END
C ***** DER2
SUBROUTINE DER2(S,W,DW)
RETURN
END
C ***** DER3 *****88
SUBROUTINE DER3(S,W,DW)
IMPLICIT REAL*8 (A-H,O-Z)
C INTEGRATE TRANSVERSE AND LONGITUDINAL ENVELOPES.
INCLUDE 'CDERIV.CCC'
EQUIVALENCE (L,DAT(1)), (GRAD,DAT(2)), (BRHO,DAT(3)),
1 (EPS,DAT(4)), (Q1,DAT(5)),
2 (KZ,DAT(8)), (EPSL,DAT(10)), (G,DAT(11))
DIMENSION W(7,1),DW(7,1)
REAL*8 K,KZ,L
C
IF (!FLAG) 1,1,2
1 BRHO=BRHO*(1.D0+ DP)
K = GRAD/BRHO
EPS2 = EPS*EPS
EPSL2 = EPSL*EPSL
Q1G = Q1*G
RETURN
C
2 AX = W(1,1)
AY = W(3,1)
AZ = W(5,1)
AX3 = AX*AX*AX
AY3 = AY*AY*AY
AZ3 = AZ*AZ*AZ
AXYZ = AZ*(AX+AY)
AZ2 = 2.D0*AZ*AZ
EAX3 = EPS2/AX3
EAY3 = EPS2/AY3
EAX3 = EPSL2/AZ3
QAXYZ = Q1/AXYZ
QAZ = Q1G/AZ2
C
DW(1,1) = W(2,1)
DW(3,1) = W(4,1)
DW(5,1) = W(6,1)
C
DW(2,1) = -K*AX + EAX3 + QAXYZ
DW(4,1) = K*AY + EAY3 + QAXYZ
DW(6,1) = KZ*AZ + EAZ3 + QAZ
C

```

RETURN
END

C ***** DER4

SUBROUTINE DER4(S,W,DW)
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'CDERIV.CCC'
EQUIVALENCE (L,DAT(1)),(GRAD,DAT(2)),(BRHO,DAT(3)),
1 (EPS,DAT(4)),(Q,DAT(5))
DIMENSION W(4,1),DW(4,1)
REAL*8 L,K,KX,KY,KBX,KBY

IF (IFLAG) 1,1,2
1 BRHO = BRHO*(1.+DP)
K = GRAD/BRHO
EPS2 = EPS*EPS
RETURN

C ENVELOPE EQUATIONS

2 AX=W(1,1)
AY=W(3,1)
AX3=AX*AX*AX
AY3=AY*AY*AY
A=AX+AY
EAX3=EPS2/AX3
EAY3=EPS2/AY3
QA=Q/A

DW(1,1) = W(2,1)
DW(3,1) = W(4,1)
DW(2,1) = -K*AX + EAX3 + QA
DW(4,1) = K*AY + EAY3 + QA

GO TO (7,3,3,5), IFLAG

C LINEARIZED ENVELOPE EQUATIONS

3 EAX4=3.*EAX3/AX
EAY4=3.*EAY3/AY
QA2=QA/A
KBX=-K-EAX4-QA2
KBY= K-EAY4-QA2

DO 4 J=2,NR
DW(1,J) = W(2,J)
DW(3,J) = W(4,J)
DW(2,J) = KBX*W(1,J) - QA2*W(3,J)
4 DW(4,J) = KBY*W(3,J) - QA2*W(1,J)

7 RETURN

C SINGLE PARTICLE EQUATIONS

5 QAX=QA/AX
QAY=QA/AY
KX=-K+QAX
KY= K+QAY


```

        DO 6 J=2,NR
        DW(1,J) = W(2,J)
        DW(3,J) = W(4,J)
        DW(2,J) = KX*W(1,J)
6      DW(4,J) = KY*W(3,J)

        RETURN
        END
C ***** DER5
      SUBROUTINE DER5(S,W,DW)
      IMPLICIT REAL*8 (A-H,O-Z)

C
C      DIFFERENTIAL EQUATION OF THE MOTION IN A SEXTUPOLE
C
      INCLUDE 'CDERIV.CCC'
      DIMENSION W(4,1),DW(4,1)
      REAL*8 K

C
      IF (IFLAG) 1,1,2
1      BRHO=(1.+DP)*DAT(3)
      K=DAT(2)/BRHO
      RETURN

C
2      DW(1,1)=W(2,1)
      DW(2,1)=-K*(W(1,1)*W(1,1)-W(3,1)*W(3,1))/2.
      DW(3,1)=W(4,1)
      DW(4,1)=K*W(1,1)*W(3,1)

      IF(IFLAG.EQ.1) RETURN

      DO 10 J=2,5
      DW(1,J)=W(2,J)
      DW(2,J)=-K*(W(1,1)*W(1,J)-W(3,1)*W(3,J))
      DW(3,J)=W(4,J)
10     DW(4,J)=K*(W(1,1)*W(3,J)+W(3,1)*W(1,J))

C
      RETURN
      END
C ***** DER6
      SUBROUTINE DER6(S,W,DW)
      RETURN
      END
C ***** DER7
      SUBROUTINE DER7(S,W,DW)
      RETURN
      END
C ***** DIFEQ
      SUBROUTINE DIFEQ(NN,V,DERI)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      DIMENSION T(8,28)
      REAL*16 H,T
      LOGICAL SWPR,OUT
      DIMENSION V(7,1),W(28),DW(28)

```

```
EXTERNAL DERI
INCLUDE 'CDERIV.CCC'
```

```
C INTEGRATE FIRST ORDER DIFFERENTIAL EQUATIONS USING ZAM.
C S=INDEPENDENT VARIABLE, GOES FROM 0 TO SMAX.
C H0=MAXIMUM INTEGRATION STEP SIZE (WILL BE ADJUSTED TO BE AN INTEGRAL
C FRACTION OF SMAX).
C NPR=NUMBER OF INTERVALS H0 BETWEEN PRINTING OF S,W.
C W= DEPENDENT VARIABLE VECTOR, DW=DERIVATIVE VECTOR.
C DERI IS THE NAME OF THE SUBROUTINE THAT CALCULATES DW(I).
C ITS FORM IS — SUBROUTINE DERI(S,W,DW)
```

```
NDAT=INFF(4,NN)
NPR=INFF(5,NN)
NF = INFF(15,NN)
```

```
MN = MR*NR
CALL DATA (NN,1,1,NF,DAT)
NAMOP = INFF(1,NN)
IF (NAMOP.NE.3HDRF) GO TO 1
```

```
DAT2(2) = 0.
DAT2(3) = 1.D0
DAT2(4) = 0.
```

```
1 SMAX = DAT(1)
H0 = DAT(6)
IF (NAMOP.EQ.3HELQ) CALL SETALPH(NN)
IF (MP2FLG.EQ.1) H0 = DAT2(7)
IF (H0.EQ.0.) H0 = .01D0
```

```
DO 2 J=1,NR
JV=NV(J)
JM = MR*(J-1)
```

```
DO 2 I=1,MR
2 W(I+JM) = V(I,JV)
```

```
JV1=NV(1)
DP = V(6,JV1)
```

```
S = 0.
ISTEP=0
IOSTP=0
NSTP=SMAX/H0+.000001D0
H=SMAX/DFLOAT(NSTP)
OUT=.FALSE.
IF (NPR.GT.0) OUT=.TRUE.
```

```
IFLAG=-IFLAG
UZ = 0.
CALL DERI(S,W,DW)
W(2) = W(2) + UZ
IFLAG = -IFLAG
CALL ZAM0(MN,S,DERI,W,DW,T,H)
```

```
IF (.NOT.OUT) GO TO 3
WRITE(3,101) ISTEP,S, (W(I),I=1,MN)
```

```
3 CALL ZAM(S,DERI,W,DW,T,SWPR)
```

```
    ISTEP=ISTEP+1
    IF (SWPR) GO TO 4
    GO TO 5
```

```
4 IF (.NOT.OUT) GO TO 5
  IOSTP=IOSTP+1
  IF (IOSTP.LT.NPR) GO TO 5
  WRITE(3,101) ISTEP,S, (W(I),I=1,MN)
  IOSTEP=0
```

```
5 IF (S.LT.SMAX-.0000000001D0) GO TO 3
  W(2) = W(2) - UZ
  IF (.NOT.OUT) GO TO 7
  WRITE(3,101) ISTEP,S, (W(I),I=1,MN)
```

```
7 DO 6 J=1,NR
  JV=NV(J)
  JM = MR*(J-1)
  DO 6 I=1,MR
6   V(I,JV) = W(I+JM)
```

```
    RETURN
101 FORMAT(5X,I5,10F11.6/(10X,10F11.6))
    END
```

```
C ***** DET *****
```

```
    SUBROUTINE DET( C, X )
    IMPLICIT REAL*8 (A-H,O-Z)
```

```
C CALCULATE THE DETERMINANT OF A 3X3 COMPLEX MATRIX.
```

```
    COMPLEX*16 C,X
    DIMENSION C(3,3)
    X = C(1,1)*( C(2,2)*C(3,3) - C(2,3)*C(3,2) ) -
1      C(1,2)*( C(2,1)*C(3,3) - C(2,3)*C(3,1) ) +
2      C(1,3)*( C(2,1)*C(3,2) - C(2,2)*C(3,1) )
    RETURN
    END
```

```
*****DOIT*****
```

```
    SUBROUTINE DOIT(MSR)
    IMPLICIT REAL*8 (A-H,O-Z)
```

```
    INTEGER OPNAME,BDAT
    INCLUDE 'BINFF.CCC'
    INCLUDE 'BMI.CCC'
    INCLUDE 'BMI1.CCC'
    INCLUDE 'CONTRL.CCC'
    INCLUDE 'MATCH3.CCC'
    INCLUDE 'MOVTOG.CCC'
    INCLUDE 'TSW.CCC'
    INCLUDE 'SWTCH.CCC'
```

```
    EQUIVALENCE(INDEF,IINDEF)
    DIMENSION RW(3),V0(7),T(50),PAR(50),KREM(50),MIM(100)
```

COMMON/MIM/MIM
EQUIVALENCE (PAR,KREM)

DIMENSION N(200),NEQ(15)
EQUIVALENCE (N(11),NEQ(1))
DATA N(5),N(51),N(161),N(102),N(104)
1 /1,17,18,19,20/
DATA N(47),N(78) /21,22/
DATA N(28),N(123)/23,24/
DATA NEQ /2,3,4,5,6,7,8,9,10,11,12,13,14,15,16/
JMT = 0
MS = MSR - 1

100 OPNAME = INFF(1,MS)
IF (ERROR) RETURN -
IOPMS = INFF(3,MS)

C SCALAR OPERATIONS — GO TO 76
IF (IOPMS.GE.112.AND.IOPMS.LE.122) GO TO 76
IF (IOPMS.LT.0) GO TO 200
NSTAT=N(IOPMS)
IF(NSTAT.EQ.IINDEF) GO TO 201

C GO TO (END,=,CRD,BML,DRF,MAG,MAGV,EQU,INV,SHF,
C REF,*,MMM,CYA,CYB,CYC,SUM,TRKB,TRKE,TRKM,VEC,MXV,
C SHF7,MOVE,TAB,CALC)

GO TO (5,11,200,200,14,15,16,200,18,19,
1 20,21,22,201,201,23,51,75,176,177,200,104,
2 47,78,810,77), NSTAT

5 RETURN

C ERROR ENCOUNTERED. OPERATION CALLED IS NOT ALLOWED.

201 WRITE(3,1) OPNAME
ERROR = .TRUE.
1 FORMAT (13H ***ERROR*** A5,57H IS AN OPERATION THAT CANNOT BE CALL
1ED BY SUBROUTINE DOIT)
RETURN

C DRIFT ENTRY

14 KA = INFF(4,MS)
KB = INFF(5,MS)
IF (KA.GT.0) GO TO 1033
EL = FDAT(MS,1)
IF (KB.EQ.0) GO TO 1066

C VARIABLE DRIFT LENGTH

FLTOT = 0.
DO 1044 K=1,KB
MM = MDAT(MS,K)

1044 FLTOT = FLTOT + FDAT(MM,1)
FLSEP = FDAT(MS,2)
IF (FLSEP.NE.0.) GO TO 1055
FLSEP = EL + FLTOT
CALL REPFLT (MS,2,FLSEP)

```

      GO TO 1066
1055 EL = FLSEP - FLTOT
      CALL REPFLT(MS,1,EL)
1066 CALL DRIFT(MS,EL)
      GO TO 200
1033 CALL DRFMAT(MS,KA)
      GO TO 200

```

C MAGNET ENTRY

```

15  KA = INFF(4,MS)
      IF (KA.GT.0) GO TO 130
      CALL DATA (MS,1,1,6,PAR)
      CALL MAGNET (MS,PAR)
      GO TO 200
130 CALL MAGMAT(MS,KA)
      GO TO 200

```

C MAGV ENTRY

```

16  CALL DATA (MS,1,1,6,PAR)
      CALL MAGV(MS,PAR)
      GO TO 200

```

C

C MMM ENTRY

```

22  NDAT = INFF(17,MS)
      CALL MIFILL (MS,1,NDAT,NELM,MIM)
      KB = INFF(5,MS)
      IF (KB.NE.0.AND.KB.NE.1) GO TO 142
      CALL MMM(MS,NELM,MIM)
      GO TO 200
142 CALL EXECMM(MS,NELM,KB,MIM)
      GO TO 200

```

C MOVE ENTRY

```

78  DO 781 I=1,7
781 V0(I) = 0.
      CALL MOVE(MS,MS,V0)
      TOG = .TRUE.
      GO TO 200

```

C

C MXV ENTRY

```

104 M1 = MDAT(MS,1)
      M2 = MDAT(MS,2)
      LOC = 1
      IF (.NOT.VCSW) GO TO 105
      DECODE (10,106,M2) NUM
106 FORMAT (1X,11,8X)
      LOC = (NUM-1)*7 + 1
105 CALL DATA (M2,5,LOC,7,PAR(8))
C      = ENTRY
      CALL RTRV7(M1,T,RW)
      CALL MXV7(T,PAR(8),PAR(1))
      CALL STDAT(MS,5,1,7,PAR)
      GO TO 200

```

C REF ENTRY

```

20  NJ = MDAT(MS,1)

```

```

      CALL REF(MS,NJ)
      GO TO 200
C   SHF ENTRY
  19   CALL DATA(MS,1,1,4,PAR)
      CALL SHF(MS,PAR)
      GO TO 200
C   SHF7 ENTRY
  47   CALL DATA (MS,1,1,6,PAR)
      CALL SHF7 (MS,PAR)
      GO TO 200
C
C   INV ENTRY
  18   NJ = MDAT(MS,1)
      CALL INV(MS,NJ)
      GO TO 200
C   STAR ENTRY
  21   KS = INFF(4,MS)
      CALL STAR(MS,KS)
      GO TO 200
C   SUM ENTRY
  51   KS = INFF(4,MS)
      CALL DATA(MS,1,1,KS,PAR)
      CALL SUM(KS,PAR,RES)
      CALL STDAT(MS,5,1,1,RES)
      GO TO 200
C   = ENTRY
  11   KN = INFF(20,MS)
      IF (KN.NE.4HSCAL) GO TO 266
      IOP = BDAT(MS,1)
      IF (IOP.EQ.1H ) GO TO 200
      IQ = 2
      ITP = INFF(21,MS)
      IF (ITP.NE.4HSNGL) IQ = 4
      CALL DATA(MS,1,1,IQ,PAR)
      CALL AROP(MS,PAR,IOP)
      GO TO 200
  266  KREM(1)=BDAT(MS,1)
      KREM(2)=BDAT(MS,3)
      IOP = BDAT(MS,2)
      CALL AROP(MS,KREM,IOP)
      GO TO 200
C   SCALAR OPERATIONS
  76   XX = FDAT(MS,1)
      NOPMS = IOPMS - 111
      GO TO (86,87,88,89,90,91,92,93,94,95,96) NOPMS
  86   YY = DSIN(XX)
      GO TO 99
  87   YY = DCOS(XX)
      GO TO 99
  88   YY = DEXP(XX)
      GO TO 99
  89   YY = DLOG(XX)
      GO TO 99

```

```

90  YY = XX*XX
    GO TO 99
91  YY = DSQRT(XX)
    GO TO 99
92  YY = DTAN(XX)
    GO TO 99
93  YY = DASIN(XX)
    GO TO 99
94  YY = DACOS(XX)
    GO TO 99
95  YY = DATAN(XX)
    GO TO 99
96  YY = DABS(XX)
99  CALL STDAT(MS,5,1,1,YY)
    GO TO 200

```

* CYA AND CYC ENTRIES

```

23  CONTINUE
25  NDAT = INFF(17,MS)
    KA = INFF(4,MS)
    IF (MINFLG.NE.4.OR.MIFLG)  CALL MIFILL(MS,1,NDAT,NELS1,MI1)
    CALL CYX(MS)
C    GO TO 280
    GO TO 200

```

* CYB ENTRY

```

C 24  CALL SETCYB(MS)
C    CALL CYX(MS)
C 280 JMT = JMT + 1
*    IF TRSW IS ON, ANGLE FIT WILL BE REPLACED BY TRACE FIT
C 122 IF (TRSW) RETURN
C    GO TO 200

```

C TRKB ENTRY

```

75  CALL TRKB(MS)
    GO TO 200

```

C TRKE ENTRY

```

176 CALL TRKE(MS)
    GO TO 200

```

C TRKM ENTRY

```

177 CALL TRKM(MS)
    GO TO 200

```

C TAB ENTRY

```

810 TABSW = .TRUE.
    CALL TABULT(MS,TABSW)
    GO TO 200

```

C CALC ENTRY

```

77  CALL CALC(MS)
    GO TO 200
200 NXTM = INFF(8,MS)
    MS = NXTM

```

GO TO 100
END

C ***** DRFMAT *****

SUBROUTINE DRFMAT(M,KA)
IMPLICIT REAL*8 (A-H,O-Z)

C FORM KA NUMBER OF DRIFT MATRICES USING ELEMENTS OF A VECTOR AS

C LENGTHS. NAME OF VECTOR REPLACED LENGTH ON DRF INPUT CARD.

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'CONTRL.CCC'

INTEGER ELNUM
INTEGER doIvec/'VEC'/,FFF0/ZFFFFFFF00/

C RETRIEVE NAME OF VECTOR FROM DRF INPUT. CHECK THAT IT IS A
C VECTOR WITH AT LEAST KA ELEMENTS.

C VECTOR NAME IS PRESUMED TO BE IN SPECIAL INTERNAL FL. PT.

C SYMBOLIC FORM — 00000NAMEV

LF = INFF(14,M)
NR=ISTORE(2*LF)
NL=LAND(FFF0,ISTORE(2*LF-1))
IF (NL.EQ.0.AND.NR.NE.0) GO TO 20

C ERROR. NOT SYMBOLIC FORM
WRITE (3,1) ISTORE(2*LF-1),ISTORE(2*LF)
GO TO 50

1 FORMAT (5X,3H***,1X,'THE NEXT VARIABLE NUMBER IS NOT ACCEPTABLE
1AS A VECTOR,','/9X,'WHOSE ELEMENTS ARE NEEDED FOR DEFINING MULTIPLE
2 DRIFTS.','2X,2A4)

C GET VECTOR NAME, INDEX AND OPNAME.

20 NMVEC=NR
MV = ELNUM(NMVEC)
NMOP = INFF(1,MV)
IF (NMOP.NE.doIvec) GO TO 40
NK = INFF(4,MV)
IF (KA.GT.NK) GO TO 45
M1 = M - 1

DO 25 I=1,KA
EL = FDAT(MV,I)

C STORE LENGTH IN FL. PT. INPUT

CALL REPFLT(M1,1,EL)
CALL DRIFT(M1,EL)

M1 = M1 - 1

25 CONTINUE

RETURN

40 WRITE (3,3) NV


```

3   FORMAT (5X,3H***,1X,A5,1X,' IS NOT A VECTOR.  CANNOT FORM MULTIPLE
1   DRIFT MATRICES. ')
    GO TO 50
45  WRITE(3,2) NV,NK,KA
2   FORMAT (5X,3H***,1X,A5,1X,'HAS ONLY'1X,I5,1X,'ELEMENTS.'/9X,I5,1X,
1   'ARE NEEDED TO DEFINE CURRENT DRIFT MATRICES. ')
50  ERROR = .TRUE.
    RETURN
    END

```

```

C ***** DRFRSV *****
    SUBROUTINE DRFRSV( M0,NAME0,KA0,KB0,NAME1 )
    IMPLICIT REAL*8 (A-H,O-Z)
C   SETS UP INFO AND STORAGE FOR A DRIFT.
C   NAME0 = NAME OF DRIFT
C   RESERVES DRF INPUT OF 1 PARAMETER, OUTPUT FOR 3X3 MATRIX
C   M0 = M INDEX, KA0 = KA, KB0 = KB
C   NAME0 = NAME      OPNAME IS DRF
C   IF M0 IS NEGATIVE, THE FIRST LETTER OF NAME1 WILL PRECEDE NAME0
C   TO FORM NAME.
    DATA MXPMY          /1      /
    INTEGER NEGZER/-32767/
    CALL STINFO( M0,NAME0,4HDRF ,KA0,KB0,NAME1 )
    IF (KB0.EQ.0) CALL RESRV8(M0,MXPMY,NEGZER,1,1,0,0,0)
    IF (KB0.GT.0) CALL RESRV8(M0,MXPMY,NEGZER,2,1,KB0,2,0)
    RETURN
    END

```

```

C ***** DRIFT *****
    SUBROUTINE DRIFT(M,EL)
C   SETS UP DRIFT SPACE MATRICES RX AND RY,DIMENSIONED (2,3)  RY=RX
C   THERE ARE 3 WAYS TO DEFINE DRIFTS —
C
C   -STANDARD METHOD, EXAMPLES —
C 2— 8— -15 -19 21—•—31—•—41—•—51—•—61—•—71
C D4   DRF          3.85
C
C  LD4  =          3.85
C  D4   DRF          LD4
C
C  D4 IS THE NAME OF A DRIFT OF LENGTH 3.85
C
C   -METHOD 2, CREATE A SET OF DRIFTS —
C 2— 8— -15 -19 21—•—31—•—41—•—51—•—61—•—71
C L   VEC          .54      2.14      3.85      22.67      25.
C D   DRF          L
C
C  5 DRIFTS ARE CREATED. D1,D2,D3,D4,D5 OF LENGTHS = .54, 2.14, —25.
C
C   -METHOD 3 IS USED TO ADJUST THE LENGTH OF THE DRIFT TO MAINTAIN THE
C   SUM OF A SET OF DRIFTS AND/OR MAGNETS AT CONSTANT LENGTH.
C 2— 8— -15 -19 21—•—31—•—41—•—51—•—61—•—71
C DDD DRF          KB LDDD      LTOT      D1   D2   —   DKB
C
C  IF LTOT .NE. 0, LDDD IS ADJUSTED TO MAKE LDDD+LD1+...+LDKB = LTOT
C  IF LTOT .EQ. 0, LTOT IS STORED AS LTOT = LDDD+LD1+...+LDKB
C  ON SUBSEQUENT CALLS, LDDD WILL BE ADJUSTED TO MAINTIAN CONSTANT LTOT.

```

```

C
  IMPLICIT REAL*8 (A-H,O-Z)
  DIMENSION RX(2,3),RY(2,3),RW(3)
  DATA RW(2),RW(3)/2*0.0/
C SETS UP DRIFT SPACE MATRICES RX AND RY,DIMENSIONED (2,3)  RY=RX
  DO 1 I=1,2
  DO 1 J=1,3
  RX(I,J)=0.0
1  CONTINUE
  RX(1,1)=1.
  RX(1,2)=EL
  RX(2,2)=1.0
  RW(1) = EL
  CALL STXY(M,RX,RX,RW)
  RETURN
  END
C ***** DUPIN *****
  SUBROUTINE DUPIN
  IMPLICIT REAL*8 (A-H,O-Z)
  INCLUDE 'FILES.CCC'
  DIMENSION ICARD(20)

  OPEN (UNIT=IFILE,STATUS='SCRATCH')
  REWIND 2
100 READ (2,20,END=200) (ICARD(I),I=1,20)
  WRITE (IFILE,20) (ICARD(I),I=1,20)
  GO TO 100
200 REWIND 2
  READ (2,20) (ICARD(I),I=1,20)
  RETURN
20  FORMAT (20A4)
  END

C ***** EDRF *****
  SUBROUTINE EDRF(NBSY,Y,NBP)
  RETURN
  END

C ***** ELNUM *****
  INTEGER FUNCTION ELNUM(NAME)
  IMPLICIT REAL*8 (A-H,O-Z)
C SEARCHES FOR INDEX OF ELEMENT NAMED NAME
  INCLUDE 'BINFF.CCC'
  INCLUDE 'STORE.CCC'
  INCLUDE 'SWTCH.CCC'
  INCLUDE 'FLTN.CCC'
  INCLUDE 'CONTRL.CCC'
  INCLUDE 'MATCH.CCC'

C LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF
  DATA IBLNK/'  '/
  DATA MASK /Z000000FF/
C  DATA FOR SPACE
  DATA IFILL/Z20000000/
  LOGICAL MINV
  DIMENSION INB(10)

```

```

DATA INB/'0','1','2','3','4','5','6','7','8','9'/
INTEGER doIPVEC/'PVEC'/,doICVEC/'CVEC'/
INTEGER SHFTL, SHFTR
INTEGER doICYC/'CYC'/,doICYA/'CYA'/

MINV=.FALSE.
VCSW=.FALSE.
MSK=LAND(MASK,NAME)
MSK = SHFTL( MSK, 24)
MSK1=LAND(MASK,IBLNK)
MSK1 = SHFTL( MSK1, 24)
IF(MSK.EQ.MSK1) GO TO 30
IF (MSK.NE.IFL(15)) GO TO 15
MINV=.TRUE.
LAG=SHFTR(NAME,8)
NAM=LOR(LAG,IFILL)
GO TO 16
15  NAM = NAME
16  NM = NAM
17  DO 20 I=MIN,MAX
    IF (NM.EQ.INFF(2,I)) GO TO 25
20  CONTINUE
C  IF MODE IS 2, ELEMENT NOT YET DEFINED. SET ELEMENT NO. TO 0,RETURN
    IF (MODE.NE.2) GO TO 22
    ELNUM = 0
    GO TO 40
22  IF(VCSW) GO TO 23
C  IF SECOND LETTER OF NAME IS NOT A NUMBER, STOP SEARCHING.

    NNAM=SHFTR(NAM,8)
    INAM=LOR(NNAM,IFILL)

    DO 21 I=1,10
        IF(INAM.EQ.INB(I)) GO TO 24
21  CONTINUE
    GO TO 23
24  CONTINUE
C  SEARCH FOR ELEMENT WHOSE WHOLE NAME IS FIRST LETTER OF THIS ONE.
    VCSW = .TRUE.
    DECODE(4,2,NAM)NM
2  FORMAT (A1,3X)
    GO TO 17
C  FAILED TO MATCH NAMES
23  ERROR = .TRUE.
    ELNUM = -32767
    WRITE(3,11) NAME
    GO TO 40
11  FORMAT(15H *** ERROR *** ,A5,30H IS AN UNDEFINED ELEMENT NAME.)
25  II=I
    IF (.NOT.VCSW) GO TO 26
    NOP=INFF(1,II)
    IF ( (NOP.EQ.doICYC).OR.(NOP.EQ.doICYA) ) GO TO 26
    IF ( (NOP.NE.doIPVEC).AND.(NOP.NE.doICVEC) ) GO TO 23
26  ELNUM = II
    IF (MODE.EQ.2) GO TO 40

```

```

      IF(.NOT.MINV) GO TO 40
      ELNUM = - ELNUM
      GO TO 40
30    ELNUM=-32767
      IF (ERPR) WRITE(3,31) NAME
31    FORMAT (15H *** ERROR *** ,A5,' IS NOT A LEFT ADJUSTED ELEMENT NAM
1E')
      ERPR=.TRUE.
40    CONTINUE
      RETURN
      END
C*****8
      SUBROUTINE ELQ(LOC,MM,V0,L,W,E)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BSTORE.CCC'

C      COMMON PI,R,TH,ER,ET
      REAL L
      DIMENSION AL(1),BI(4)
      DIMENSION W(3), E(4)
      DATA PI/3.1415926536D0/

C
      X=W(1)
      Y=W(2)
      Z=W(3)
      R=DSQRT(X*X+Y*Y)
      TH=DATAN2(Y,X)
      V0L=PI*V0/L
      V0R=2.D0*V0/R
      RL=PI*R/L
      ZL=PI*Z/L
      V=0.
      ER=0.
      ET=0.
      EZ=0.
      TH2=2.D0*TH
      CT=DCOS(TH)
      ST=DSIN(TH)
      CT2=DCOS(TH2)
      ST2=DSIN(TH2)

C
      DO 1 M=1,MM
      ALM = STORE(LOC+M)
      FM=2*M-1
      BLM=ALM*FM
      RM=FM*RL
      ZM=FM*ZL
      SZ=DSIN(ZM)
      CZ=COS(ZM)
      CALL BESIS(RM,BI,3)
      V = ALM*BI(3)*SZ + V
      ER = BLM*(BI(2)+BI(4)) *SZ + ER
1    EZ = BLM*BI(3)*CZ + EZ

C
      ER = -(V0L/2.D0)*CT2*ER

```

```

      ET = V0R*ST2*V
      EZ = -V0L*CT2*EZ
      V = V0*CT2*V
C
      EX = ER*CT - ET*ST
      EY = ER*ST + ET*CT
C
      E(1)=EX
      E(2)=EY
      E(3)=EZ
      E(4)=V
      RETURN
      END
C ***** EMIS *****
      SUBROUTINE EMIS
      IMPLICIT REAL*8 (A-H,O-Z)
C  END MISALIGNMENT
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'SWTCH.CCC'

      INTEGER IBLNK/1H /,doIMIN/'-' /
C
      IF (MODE.EQ.3) GO TO 10
      ISGN=IBLNK
      IF (OP .LT. 0) ISGN=doIMIN
      WRITE (3,21) ISGN
21  FORMAT (6H *** ,A1,8X,4HEMIS)
      IF (OP .LT. 0) RETURN
      IF (MODE.EQ.2) RETURN
10  BEND =1.
      RETURN
      END
C ***** EMIT *****
      SUBROUTINE EMIT(M)
      IMPLICIT REAL*8 (A-H,O-Z)
C  COMPUTES EMITTANCES FROM NORMALIZED INPUT

      DIMENSION PAR(4),EM(3)
      EQUIVALENCE (EPSX,EM(1)),(EPSY,EM(2))
      DATA E0/.9382592D0/

      CALL DATA (M,1,1,4,PAR)
      BETGAM = PAR(1)/E0
      EPSX = PAR(2) /BETGAM
      EPSY = PAR(3) /BETGAM
      EM(3) = PAR(4)
      CALL STDAT(M,5,1,3,EM)
      RETURN
      END
C ***** ENVTRK *****
      SUBROUTINE ENVTRK
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'

```

```

INCLUDE 'TRKINT.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'BMI.CCC'

```

```

RETURN

```

```

END

```

```

C ***** EQIL *****

```

```

SUBROUTINE EQIL(M)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

C   CALCULATE 3 CLOSED ORBITS.SYNCHRONOUS,DP/P=0,DP/P=1.
C   M IS A 7X7 MATRIX PARTITIONED AS FOLLOWS.

```

```

INCLUDE 'CONTRL.CCC'

```

```

DIMENSION T(49),RW(3),TN(6,7),TT(7,7),V(7,3),U(7,3)
EQUIVALENCE (T(1),TT(1))

```

```

INTEGER BDAT

```

```

INTEGER doINOPR/'NOPR'/,doINOSV/'NOSV'/

```

```

M1 = MDAT(M,1)

```

```

IOP = BDAT(M,2)

```

```

CALL RTRV7(M1,T,RW)

```

```

IF (ERROR) RETURN

```

```

N = 6

```

```

DO 25 K=1,3

```

```

JJ = 7

```

```

IF (K.EQ.3) JJ= 6

```

```

DO 10 J = 1,N

```

```

DO 10 I = 1,N

```

```

TN(I,J) = TT(I,J)

```

```

10 CONTINUE

```

```

DO 15 I = 1,N

```

```

TN (I,I) = TN(I,I) -1.

```

```

TN(I,N+1) = -TT(I,JJ)

```

```

15 CONTINUE

```

```

CALL MXDIV(TN,N,1,D,6)

```

```

DO 20 I=1,N

```

```

V(I,K) = TN(I,N+1)

```

```

20 CONTINUE

```

```

N = 4

```

```

25 CONTINUE

```

```

V(7,1) = 1.

```

```

MN = MNAME(M)

```

```

MT = MNAME(M1)

```

```

V(7,2) = 1.

```

```

V(6,3) = 1.

```

```

V(5,2) = 0.

```

```

V(6,2)= 0.

```

```

V(5,3) = 0.

```

```

V(7,3) = 0.

```

```

IF (IOP.EQ.doINOPR) GO TO 27

```

```

CALL DASH

```

```

C PRINT VECTORS BEFORE MULTIPLYING BY MATRIX

```

```

WRITE (3,100) MN,MT

```

```

        WRITE (3,105) ((V(I,J),I=1,7),J=1,3)
C  MULTIPLY BY MATRIX
27  DO 30 J=1,3
    CALL MXV7(TT,V(1,J),U(1,J))
30  CONTINUE
    IF (IOP.EQ.doINOPR) GO TO 32
C  PRINT VECTORS AFTER MULTIPLICATION
    WRITE (3,110) ((U(I,J),I=1,7),J=1,2)
    WRITE (3,115) MT,(U(I,3),I=1,7)
32  IF (IOP.EQ.doINOSV) RETURN
    CALL DASH
    CALL STDAT(M,5,1,21,V)
    RETURN
100  FORMAT (/20X,15H CLOSED ORBITS A6, 12H DEFINED BY A6/29X,1HX,10X,
1  5HDX/DS,8X,1HY,10X,5HDY/DS,7X,4HS-S0,8X,4HDP/P,7X,6HMISVAR//)
105  FORMAT (25H SYNCHRONOUS C.O. VS ,7(F10.5,1X) /25H DP/P = 0.
1  C.O. VO ,7(F10.5,1X)/25H DP/P = 1. C.O. VP ,7(F10.5,1X)/
2/)
110  FORMAT (25H VALUES AFTER VS ,7(F10.5,1X)/25H MULTIPLYIN
1G VO ,7(F10.5,1X))
115  FORMAT (5X,3H BY,1X,A4,7X,2HVP,3X,7(F10.5,1X))
    END
C ***** EQU *****
    SUBROUTINE EQU(M,N)
    IMPLICIT REAL*8 (A-H,O-Z)
C  EQUATES DATA STORAGE OF M TO THAT OF N.
C  * IS A FLAG TO LEAVE PRESENT VALUES OF M UNTOUCHED,
C  I.E KEEP THE ORIGINALS SET FOR M. OTHERS WILL BE
C  CHANGED TO THOSE READ IN FROM N OR RESET.
    DIMENSION INF(24)
    INTEGER doIASTRK/4H* /
    CALL RDINF(INF,N)
    DO 1 I=1,9
1  INF(I) = doIASTRK
C  USE LQ AND NQ OF N. COMPUTE NEW NTOT.
    DO 2 I=12,17
2  INF(I) = doIASTRK
    INF(18) = INF(11) + INF(17)
C  USE KIND,ROWS,ETC OF N
C  REWRITE INFO OF M
    CALL WRTINF(INF,M)
    RETURN
    END
C ***** ERRSR *****
    SUBROUTINE ERRSR(M)
    IMPLICIT REAL*8 (A-H,O-Z)
C  ERROR HAS BEEN ENCOUNTERED. EXECUTION OF RUN HALTED.
C  SEARCHES FOR AN INPUT WFL. IF IT EXISTS WITH KA=0, CALLS WFLSR.
C  READS CARDS TO FIND END OF RUN
C  RETURNS TO MAIN FOR NEXT RUN
    INCLUDE 'BINFF.CCC'
    INCLUDE 'STORE.CCC'
    INTEGER OP,OPNAME,OPNUM
    INTEGER doIFIN/'FIN'/
C  SEARCH FOR A WFL

```

```

      NM = 3HWFL
      DO 50 MM = MIN,MAX
      INAM = INFF(1,MM)
      IF (NM.EQ.INAM) GO TO 60
50    CONTINUE
C   FAILED TO FIND A WFL
      GO TO 1
C   FOUND A WFL
60    IOP = INFF(3,MM)
      IF (IOP.LT.0) GO TO 1
      KA = INFF(4,MM)
      IF (KA.EQ.0) CALL WFLSR(MM)
1    READ (2,5) (ICARD(IW),IW=1,20)
5    FORMAT(20A4)
      DECODE(12,2,ICARD)OPNAME
2    FORMAT(7X,A4,1X)
      IF (OPNAME.NE.doIFIN) GO TO 1
      WRITE (3,20)
20   FORMAT (30H *** ERROR *** RUN TERMINATED./1X,119(1H.))
      RETURN
      END
C*****EVEC*****
      SUBROUTINE EVEC(M)
      RETURN
      END
C ***** EXCHM *****
      SUBROUTINE EXCHM(M,N)
      IMPLICIT REAL*8 (A-H,O-Z)
C   RX(N) IS SET EQUAL TO RY(M),AND RY(N) IS SET EQUAL TO RX(M)
      DIMENSION RX(2,3),RY(2,3),RW(3)
      CALL RXY(M,RX,RY,RW)
      CALL STXY(N,RY,RX,RW)
      RETURN
      END
C ***** EXECMM *****
      SUBROUTINE EXECMM(M,NELS,KB,MI)

C   EXECUTES MMM (MATRIX MULTIPLICATION) INSTRUCTION
C   MATRIX MULTIPLICATION.  R(M)=R(MK)*...R(M2)*R(M1)
C   IF MI(I) IS NEGATIVE, R(M)=R(MK)...(1/R(MI))..R(M2).R(M1) ETC.

C   SYNCH INSTRUCT MMM
C   X   MMM   KB // A   B   C   .   .   .
C   A,B,C,... ARE PREVIOUSLY DEFINED MATRICES, OR BML'S, OR NUMBERED
C   PARENTHESES.  THIS INPUT LIST IS EXPANDED BY MIFILL AND MIEXP TO
C   FORM A LIST OF MATRIC INDICES, STORED IN MI(I).  THESE ARE
C   MULTIPLIED TOGETHER TO FORM MATRIX NAMED X.

C   IF KB IS NEGATIVE, A MATRIX CORRESPONDING TO THE REFLECTION OF THE
C   INPUT BEAM LINE IS ADDED TO THE BEAM LINE.  THUS ONLY HALF OF THE
C   SYMMETRIC SUPERPERIOD NEED BE SPECIFIED.

C   THE ABS(KB) IS THE NUMBER OF SUPER PERIODS.  IF ABS(KB) .GT. 1, THE
C   MATRIX IS RAISED TO THAT POWER.

```



```

      INCLUDE 'DIM.CCC'

      DIMENSION MI(1)

      CALL MMM(M,NELS,MI)
      IF (KB.GT.1) GO TO 5
      CALL REF(M7END,M)
      MI(1) = M
      MI(2) = M7END
      CALL MMM(M,2,MI)
5      KK = JIABS(KB)
      IF (KK.LE.1) RETURN
      DO 10 I=1,KK
10     MI(I) = M
      CALL MMM(M,KK,MI)
      RETURN
      END

C ***** EXTGIN *****
      SUBROUTINE EXTGIN(PINT)
      IMPLICIT REAL*8 (A-H,O-Z)
CC      TRANSFORMS THE EXTERNAL PARAMETER VALUES X TO INTERNAL
CC      VALUES IN THE DENSE ARRAY PINT.  FUNCTION PINTF IS USED.
CC
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      DIMENSION PINT(2)
      LIMSET=0
      DO 100 I= 1, NU
      J = LCORSP(I)
      IF ( J ) 100,100,50
50     PINT(J) = PINTF(U(I),I)
100    CONTINUE
      RETURN
      END

C ***** FCN *****
      SUBROUTINE FCN(NPAR,G,F,X,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'MATCH.CCC'
      INCLUDE 'PAREXT.CCC'
      INCLUDE 'MATCH3.CCC'

      LOGICAL BETFLG,BXBYFLG
      DIMENSION X(30),BETX(9),BETY(9),T(7,7),RW(3),NPS(6)
      DATA NN,IBLNK/6,1H /

C  EVALUATION OF F, IE CHISQUARE, FOR MINUIT

      MINFLG=IFLAG
      NGO=2
      IF (IFLAG.EQ.4) GO TO 4
      NGO=1
      NOPR=.FALSE.

```

```

      IF (IFLAG.EQ.1.AND.IPP.NE.0) NOPR = .TRUE.
      IF (IFLAG.EQ.3.AND.IPP.GT.1) NOPR=.TRUE.
      NONU=.FALSE.
      N=0
      DDACOS = DACOS(0.D0)
      RTP1 = 1.D0/(4.D0*DDACOS)
4     CONTINUE
C   PUT PARAMETER VALUES IN DATA ARRAYS
      BETFLG = .FALSE.
      NVAR=NPARG
      DO 1 J=1,NVAR
      MEL=MPAR(J)
      INTEL=IPAR(J)
      CALL REPFLT(MEL,INTEL,X(J))
1     CONTINUE
C   CALL SYNCH SUBROUTINE
      CALL DOIT(MSR)
C   CALCULATE CHISQUARE
      GO TO (8,9) NGO
8     WRITE(3,100)
9     F = 0.0

      DO 2 K=1,KAT
      ITYPK=ITYPE(K)
      IF(ITYPK.EQ.-1) GO TO 2
      NPOSK=NPOS(K)
      ICONK=ICON(K)
      BXBYFLG = .FALSE.
      IF (ICONK.EQ.37) BXBYFLG = .TRUE.
      WTK=WT(K)
      BFITK=BET0(K)
      IF ((ITYPK.EQ.1).OR.(ITYPK.EQ.2)) GO TO 10
      IF (ICONK.EQ.34) GO TO 22
      IF (MATFLG.OR.ICONK.EQ.35) GO TO 3
      IF (BXBYFLG) GO TO 21
      LTRTBK = LTRTAB - 1 + ICONK
      GO TO 28
21     LBXK = LTRTAB + 1
      LBYK = LTRTAB + 7
28     CONTINUE

      IF(NPOSK.EQ.1) GO TO 17
      FIK = 0.0

      DO 6 L=1,NPOSK
      IPLK = 14*IPOS(L,K)
      IF (.NOT.BXBYFLG) GO TO 25
      NIKL1 = LBXK + IPLK
      NIKL2 = LBYK + IPLK
      BX = STORE(NIKL1)
      BY = STORE(NIKL2)
      BETKL = BX/BY
      GO TO 26
25     NIKL = LTRTBK + IPLK
      BETKL = STORE(NIKL)

```

```

26  DEL = BETKL - BFITK
    FIK = FIK + DEL*DEL
    GO TO (11,6) NGO
11  NTYP=5H
    N=N+1
    WRITE(3,101) N,NAMBFC(ICONK),NTYP,IPRPOS(L,K),BETKL,BFITK,DEL,
1    SIG(K)
6   CONTINUE

    GO TO 5

17  CONTINUE
    IPLK = 14*IPOS(1,K)
    IF (.NOT.BXBYFLG) GO TO 27
    NIKL1 = LBXK + IPLK
    NIKL2 = LBYK + IPLK
    BX = STORE(NIKL1)
    BY = STORE(NIKL2)
    BET1K = BX/BY
    GO TO 23
27  NIK1 = LTRTBK + IPLK
    BET1K=STORE(NIK1)

23  CONTINUE
    DEL=BET1K-BFITK
    FIK = DEL*DEL
    GO TO (12,5) NGO
12  NTYP=5H
    N=N+1
    IF (NPOSK.EQ.1) GO TO 36
    WRITE(3,104) N,NAMBFC(ICONK),NTYP,(IPRPOS(L,K),L=1,NN),BET1K,
1    BFITK,DEL,SIG(K)
    GO TO 5
36  WRITE(3,101) N,NAMBFC(ICONK),NTYP,IPRPOS(1,K),BET1K,BFITK,DEL,
1    SIG(K)
    GO TO 5

10  CONTINUE
    FIK=0.0
    DO 14 L=1,NPOSK
    NIKL=LRTAB+14*IPOS(L,K)-1+ICONK
    BETKL=STORE(NIKL)
    IF (L.EQ.1) GO TO 140
    DBKJ = BETKL - BETKJ
    DEL = DBKJ - BFITK
    FIK = FIK + DEL*DEL
140 IF (ITYPK.EQ.2) BETKL = -BETKL
    BETKJ = BETKL
    IF (L.EQ.1) GO TO 14
    GO TO (13,14) NGO
13  NTYP=5HDIF
    IF (ITYPK.EQ.2) NTYP=5HSUM
    N=N+1
    WRITE(3,101) N,NAMBFC(ICONK),NTYP,IPRPOS(L,K),DBKJ,BFITK,
1    DEL,SIG(K)

```

```

14  CONTINUE

    GO TO 5

3   CONTINUE
    IF (BETFLG) GO TO 33
    CALL BET(MTR,BETX,BETY,0)
    BETFLG = .TRUE.
33  IF (ICONK.EQ.35) GO TO 24
    IF (BXYFLG) GO TO 29
    IF (ICONK.GT.6) GO TO 31
    BET1K = BETX(ICONK)
    GO TO 32
31  BET1K = BETY(ICONK-6)
32  IF (ICONK.EQ.1.OR.ICONK.EQ.7) BET1K = RTPI * BET1K
    GO TO 23

29  BX = BETX(2)
    BY = BETY(2)
    BET1K = BX/BY
    GO TO 23

24  CONTINUE
    APQN = 0.5D0*(IPOS(1,K)*BETX(1) + IPOS(2,K)*BETY(1))*IPOS(3,K)
1   / IPOS(4,K)
    BET1K = 1.D0/ (DABS(DSIN(APQN)) + .0000000001D0)
    GO TO 23

22  CONTINUE
    CALL RTRV7(MTR,T,RW)
    II = IPOS(1,K)
    JJ = IPOS(2,K)
    BET1K = T(II,JJ)
    GO TO 23

5   F = F + FIK * WTK*WTK

2   CONTINUE

    IF (.NOT.VARWTS) GO TO 20
    DO 18 J=1,NVARS
    IF(.NOT.LVAR(J)) GO TO 18
    DIF=(X(J)-VAL(J))
    DIFWT=DIF*WTVAR(J)
    F = F + DIFWT*DIFWT
    GO TO (19,18) NGO
19  N=N+1
    SIGV=1.D0/WTVAR(J)
    WRITE(3,103) N,NAM(J),X(J),VAL(J),DIF,SIGV
18  CONTINUE

20  F = F*FNORM

    GO TO (15,16) NGO
15  WRITE(3,102) N,F

```

```

16  IF(IFLAG.NE.1) RETURN
    NOPR=.TRUE.
    NONU=.TRUE.
    DO 7 K=1,KAT
      IF (ICON(K).EQ.1) NONU=.FALSE.
      IF (ICON(K).EQ.7) NONU=.FALSE.
7   CONTINUE

    RETURN
100  FORMAT(/3X,'CONSTRAINT/',3X,'FUNCTION',13X,'POSITIONS OR INDICES'
1    13X,'PRESENT VALUE',4X,'DESIRED VALUE',4X,'ERROR',11X,
2    'TOLERANCE'/)
101  FORMAT(/3X,I5,8X,A5,2X,A5,2X,A5,30X,4F16.7)
104  FORMAT(/3X,I5,8X,A5,2X,A5,2X,6A5,5X,4F16.7)
103  FORMAT(3X,I5,8X,A10,38X,4F16.7)
102  FORMAT (/5X,27HFCN = SUM[(ERROR/TOL.)*2]/,I2,2H =,E15.7)
    END

C ***** FDAT *****
    FUNCTION FDAT(M,J)
    IMPLICIT REAL*8 (A-H,O-Z)
C  RETRIEVES ONE PIECE OF FLOATING POINT DATA FROM INDEX J OF FLOATING
C  POINT STORAGE OF ELEMENT M
C  IF DATA IS SYMBOLIC, THE VALUE (IN FL.PT.) OF THIS VARIABLE IS
C  RETRIEVED.
C  if symbol refers to a "RAND" operation, value is p1 + random*p2
    common/seed/nsr
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'TSW.CCC'
    DIMENSION ISTORE(1)
    EQUIVALENCE (STORE(1),ISTORE(1)),(FVAR,IVAR)
    COMMON/FLTN/IFL(15)
    INTEGER E,NUM
    INTEGER SHFTR,SHFTL
    LOGICAL NEG
    INTEGER FFF0 /ZFFFFFFF00/, IDOL /1H$/
    DATA MASK/ZFF000000/
    DATA MBLK/Z20000000/

    NEG=.FALSE.
    LOC = INFF(14,M) + J - 1
C  TEST FOR SYMBOLIC
    NR=ISTORE(2*LOC)
    NL=LAND(FFF0,ISTORE(2*LOC-1))

    IF (NL.EQ.0.AND.NR.NE.0) GO TO 25
C  FLOATING POINT
    FDAT=STORE(LOC)
    GO TO 29
C  SYMBOLIC. LEFT ADJUST,FIND VARIABLE AND ITS VALUE
25  IVAR=NR
C  IF DOLSIGN, LEAVE AS IS.
    IF (IVAR .NE. IDOL) GO TO 24

    FDAT=FVAR

```

```

      GO TO 29
24  CONTINUE
      MSK=SHFTL(NR,24)

      IF (MSK.EQ.IFL(1)) NEG = .TRUE.
      IF (.NOT.NEG) GO TO 26
      MSK = SHFTR(NR,8)
      IVAR = MSK.OR.MBLK

26  MN = ELNUM(IVAR)
      INDEX=INFF(10,MN)
      FDAT = STORE(INDEX)
      LOC = INDEX
      IF(NEG) FDAT=-FDAT
      if(inff(1,mn).ne.4hrand) go to 29
      md = mdat(mn,2)
      dx=fldat(md,1)
      fdat = fdat + dx*(ran(nsr)-.5)

29  CONTINUE
      LOCC = LOC
      RETURN
      END
C ***** FIXPAR *****
      SUBROUTINE FIXPAR(I2,KODE,ILAX)
      IMPLICIT REAL*8 (A-H,O-Z)
CC      REMOVES PARAMETER I2 FROM THE INTERNAL (VARIABLE) PARAMETER
CC      LIST, AND ARRANGES THE REST OF THE LIST TO FILL THE HOLE.
CC      IF KODE=0, I2 IS AN EXTERNAL NUMBER, OTHERWISE INTERNAL.
CC      ILAX IS RETURNED AS THE EXTERNAL NUMBER OF THE PARAMETER.
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON/FIX/XS(15),XTS(15),WTS(15),NPFIX,IPFIX(15)
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      DIMENSION YY(15), V1(2)
      EQUIVALENCE (V(1,1) , V1(1))
      IF (KODE) 250,50,150
C
C      . EXT PARAM NO. SPECIFIED
50  I = I2
      IF (I .GT. NU) GO TO 70
      IF (I .LT. 1) GO TO 70
60  IF (LCORSP(I)) 70,70,80
C
C      ERROR RETURN PARAM ALREADY FIXED
70  ILAX = 0
      WRITE (ISYSWR,500) I
      GO TO 300
80  LC = LCORSP(I)
      IT = LC
      LCORSP(I) = 0

```

```

      ILAX = I
      NPAR = NPAR - 1
      NPFIX = NPFIX + 1
      IPFIX(NPFIX) = I
      XS(NPFIX) = X(LC)
      XTS(NPFIX) = XT(LC)
      EPS = DABS(DIRIN(LC)) * 10.D0
      IF (ISW(2) .GE. 1) EPS = EPS + DSQRT(DABS(V(LC,LC))*UP)
      IF (EPS .LT. 1.0D-10*DABS(X(LC))) EPS = 1.0D-8*X(LC)
      WTS(NPFIX) = EPS*0.1D0
      DO 100 IK= I, NU
        IF (LCORSP(IK)) 100,100,85
85    LC = LCORSP(IK) - 1
        LCORSP(IK) = LC
        X(LC) = X(LC+1)
        XT(LC) = XT(LC+1)
        DIRIN(LC) = DIRIN(LC+1)
100  CONTINUE
      IF (ISW(2) .GT. 1) GO TO 250
      ISW(2) = 0
      GO TO 300

C          . INT PARAM NO. SPECIFIED
150  CONTINUE
      DO 200 IQ= 1, NU
        IF (LCORSP(IQ) .NE. I2) GO TO 200
        I = IQ
        GO TO 60
200  CONTINUE
      GO TO 70

C          REMOVE ONE ROW AND ONE COLUMN FROM VARIANCE MATRIX
250  KON = 0
      IF (NPAR .LE. 0) GO TO 300
      KON2 = 0
      MPAR = NPAR + 1
      DO 260 I= 1, MPAR
260  YY(I)=V(I,IT)
      DO 294 I= 1, MPAR
        IF (I.EQ.IT) GO TO 294
        KON2 = KON2 + 1
        DO 292 J= 1, MPAR
          IF (J .EQ. IT) GO TO 292
          KON = KON + 1
          V1(KON)=V(J,I) - YY(J)*YY(I)/YY(IT)
292  CONTINUE
        KON = MAXINT*KON2
294  CONTINUE

C          CHECK FOR WELL-BEHAVED FINAL MATRIX
      DO 295 I= 1, NPAR
        IF (V(I,I) .LE. 0.D0) GO TO 296
        DO 295 J= 1, NPAR
          IF (I .EQ. J) GO TO 295
          IF (V(I,J)**2 .GE. V(I,I)*V(J,J)) V(I,J) = 0.D0
295  CONTINUE
      GO TO 300
296  ISW(2) = 0

```

```

      WRITE (ISYSWR, 501)
300 RETURN
500 FORMAT ( 28H0ERROR IN FIXPAR.  PARAMETER13, 18H WAS NOT VARIABLE
1./)
501 FORMAT ( 72H0COVARIANCE MATRIX WAS ILL-CONDITIONED AND HAS BEEN D
1ESTROYED BY FIXPAR./)
      END
C***** FNBML *****
      SUBROUTINE FNBML(MN,NPN,NP1,NP2)
      IMPLICIT REAL*8 (A-H,O-Z)
C   SETS UP PARALLEL ARRAYS FOR REPLACING ELEMENTS IN A BEAM LINE.
C   MN IS ELEMENT OF RBLI,MM IS ELEMENT OF BEAM LINE WHERE
C   REPLACEMENTS WILL BE MADE. MN AND MM ARE PUT INTO NP1,NP2.
C   NPN = OF REPLACEMENTS.
C   ELEMENT NUMBERS IN ARRAY NP1 WILL BE REPLACED BY THOSE IN NP2.

      INCLUDE 'BINFF.CCC'
C   COMMON /BINFF/ INFF(24,1000)
      DIMENSION NP1(1),NP2(1)

C   GET ELEMENT OF FBLI STATEMENT
      MFN = MDAT(MN,1)
      MM = MDAT(MFN,1)
      NPN = INFF(4,MN) + 1
      NP1(1) = MN
      NP2(1) = MM
      DO 10 I = 2,NPN
      NP1(I) = MDAT(MFN,I)
10  NP2(I) = MDAT(MN,I)
      RETURN
      END

C ***** DKE *****
      SUBROUTINE DKE(NBSY,Y,NBP)
      RETURN
      END

C ***** FIT *****
      SUBROUTINE FITT(MSR,MC,MV1,IV1,PHI,KA,KB)
      IMPLICIT REAL*8 (A-H,O-Z)
C   FITTING ROUTINE CALLED BY FITQ, FITB, FITR, FITV

C   FITS   FITQ —PHASE ADVANCE OVER 2PI
C   FITB —BETATRON FUNCTIONS WHOSE INDECES ARE KA AND KB.
C   FITR —MATRIX ELEMENTS INDEXED BY KA AND KB.
C   FITV —VECTOR ELEMENTS INDEXED BY KA AND KB.

C   INPUT — NAME   FITX   KA  KB  SUB  C  QF  QD  I1  I2  A1  A2
C              WHERE FITX MAY BE FITQ, FITB,FITR,FITV
C              SUB IS NAME OF SUBROUTINE TO EXECUTE.
C   VARY THE I1-TH PARAMETER OF QF AND I2-TH OF QD, COMPUTE C,
C   COMPARE THE RESULTS ACCORDING TO X OF FIT INSTRUCTION (PHASE ADVANCE
C   OR BETATRON FUNCTIONS IF FITQ OR FITB.  MATRIX ELEMENTS OR VECTORS
C   NAMED IN KA AND KB IF FITR OR FITV.) WITH A1 AND A2.
C   ITERATE UNTIL THEY MATCH, BUT NOT MORE THAN 100 TIMES.

```



```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (ISTORE(1),STORE(1))
INCLUDE 'GRR.CCC'
INCLUDE 'CONTRL.CCC'
INCLUDE 'CONST.CCC'

DIMENSION H(2),D(2),A(2,2),B(2,2),RX(2,3),RY(2,3),S(2),PHI(5)
DIMENSION MV1(1),IV1(1),V(2),U(2),W(2),VAR(2),FC(2),F(2),G(2)
DIMENSION RW(3)
DIMENSION RXRY(12),VC(7),BXY(20)
DIMENSION Q(2)
LOGICAL IND,INF
INTEGER P1,P2,ELNUM
INTEGER doIBLANK/' '/'
INTEGER SHFTR
LOGICAL SINGL

```

C

```

SINGL=.FALSE.
IF(QSW) GO TO 99
IF((RSW).AND.(KB.NE.doIBLANK)) GO TO 99
IF(KA.GE.100.OR.KB.NE.0) GO TO 99
SINGL=.TRUE.
KB=KA
MV1(2)=MV1(1)
IV1(2)=IV1(1)
PHI(2)=PHI(1)
99  CONTINUE
Q(1) = PHI(1)
Q(2) = PHI(2)
DEL = 1.D-6
IF(.NOT.QSW) DEL=1.0D-05
HSTEP = .01
XLAM = .9

```

C RESET PARAMETERS IF INPUT

```

IF (.NOT.RSW) GO TO 60
I1=SHFTR(LAND(MASK1,KA),24) -240
I2=SHFTR(LAND(MASK2,KA),16) -240
I3=SHFTR(LAND(MASK3,KA),8) -240
J1=SHFTR(LAND(MASK1,KB),24) -240
J2=SHFTR(LAND(MASK2,KB),16) -240
J3=SHFTR(LAND(MASK3,KB),8) -240
K1 = I1*6 + (I3-1) *2 +I2
K2 = J1*6 +(J3-1)*2 + J2
GO TO 16

```

60 K1 = KA

K2 = KB

16 M1 = ELNUM(MV1(1))

M2 = ELNUM(MV1(2))

P1 = IV1(1)

P2 = IV1(2)

C LOOK UP INPUT GUESS FOR VARIABLES

VAR(1) = F0AT(M1,P1)

```

      VAR(2) = FDAT(M2,P2)
C  SAVE ORIGINAL INPUT VARIABLES
      VR1=VAR(1)
      VR2=VAR(2)
      IF (.NOT.QSW) GO TO 50
C  FIT Q CASE
      PHI(1) = DCOS(PI2 * PHI(1))
      PHI(2) = DCOS(PI2 * PHI(2))
      GO TO 55
C  OTHER FITS
50  PHI(1) = PHI(1)/100.D0
      PHI(2) = PHI(2) /100.D0
55  K=0
C  SET MAXIMUM ITERATIONS
      KI = 100
75  DO 100 I=1,2
      U(I)=VAR(I)
      V(I)=VAR(I)
100  W(I)=VAR(I)
      V(1) = V(1) + HSTEP
      IF(SINGL) V(2)=V(1)
      W(2) = W(2) + HSTEP
      DO 115 I=1,3
      GO TO (101,102,103),I
101  VC1 = V(1)
      VC2 = V(2)
      GO TO 104
102  VC1 = W(1)
      VC2 = W(2)
      GO TO 104
103  VC1 = U(1)
      VC2 = U(2)
104  CALL REPFLT(M1,P1,VC1)
      CALL REPFLT(M2,P2,VC2)
      CALL DOIT(MSR)
      IF (ERROR) RETURN
      IF(BSW) GO TO 105
      IF (VSW) GO TO 110
      IF (RSW) GO TO 112
C  FITQ
      CALL RXY(MC,RX,RY,RW)
      FC(1) = (RX(1,1) + RX(2,2)) * .5
      FC(2) = (RY(1,1) + RY(2,2)) * .5
      GO TO 106
105  CALL BET(MC,BXBY(1),BXBY(11),0)
      FC(1)=BXBY(K1)/100.D0
      FC(2) = BXBY(K2)/100.D0
      GO TO 106
110  CALL DATA(MC,5,1,7,VC)
      FC(1) = VC(K1)/100.D0
      FC(2)=VC(K2)/100.D0
      GO TO 106
112  CALL RXY(MC,RXRY(1),RXRY(7),RW)
      FC(1)=RXRY(K1)/100.D0
      FC(2)=RXRY(K2)/100.D0

```

```

106 GO TO (107,108,109),I
107 G(1) = FC(1)
    G(2) = FC(2)
    GO TO 115
108 H(1) = FC(1)
    H(2) = FC(2)
    GO TO 115
109 F(1) = FC(1)
    F(2) = FC(2)
C   IND=INDEF(F(1))
C   INF=INFIN(F(1))
C   IF(IND.OR.INF) GO TO 130
C   IND=INDEF(F(2))
C   INF=INFIN(F(2))
C   IF(IND.OR.INF) GO TO 130
115 CONTINUE
    D(1) = PHI(1) - F(1)
    D(2) = PHI(2) - F(2)
    IF (DABS(D(1)).LE.DEL.AND.DABS(D(2)).LE.DEL) GO TO 120
C   FIT NOT CLOSE ENOUGH.  ITERATE AGAIN UNLES LIMIT HAS BEEN REACHED
    K = K + 1
    IF (K.EQ.KI) GO TO 125
C   SET UP NEXT GUESS
    HD = 1./HSTEP
    A(1,1) = (G(1) - F(1))*HD
    A(1,2) = (H(1) - F(1))*HD
    A(2,1) = (G(2) - F(2))*HD
    A(2,2) = (H(2) - F(2))*HD
    IF(SINGL) GO TO 999
    DET = (A(1,1)*A(2,2)) - (A(1,2) * A(2,1))
    B(1,1) = A(2,2)/DET
    B(1,2) = -A(1,2)/DET
    B(2,1) = -A(2,1)/DET
    B(2,2) = A(1,1)/DET
    S(1) = B(1,1)*D(1) + B(1,2)*D(2)
    S(2) = B(2,1)*D(1) + B(2,2)*D(2)
    VAR(1) = VAR(1) + (S(1)*XLAM)
    VAR(2) = VAR(2) + (S(2)*XLAM)
    GO TO 75
999  VAR(1) = VAR(1) + XLAM*D(1)/A(1,1)
    VAR(2) = VAR(1)
    GO TO 75

C   FIT SATISFIED.  LEAVE NEW VARIABLES IN STORAGE
120 WRITE (3,1) P1,MV1(1),VAR(1),P2,MV1(2),VAR(2)
    RETURN
1   FORMAT (//39H PARAMETER REPLACEMENTS MADE BY FITTING/5X,I5,1X,
    14H OF ,A5,3H = ,F20.12,10X,I5,1X,4H OF ,A5,3H = ,F20.12)

C   ITERATION LIMIT REACHED.  FIT FAILED.
125 WRITE (3,7) K,VAR
    GO TO 135
7   FORMAT (39H NUMBER OF ITERATIONS REACHED LIMIT OF I5/31H CURRENT
    1 VARIABLES ARE VAR1 = F14.8,8H VAR2 = F14.8)
C130 CONTINUE

```

```

C    WRITE (3,10)
C10  FORMAT (32H ROOT IS INDEFINITE OR INFINITE.//)
C REPLACE ORIGINAL INPUT VALUES
135  CALL REPFLT(M1,P1,VR1)
      CALL REPFLT(M2,P2,VR2)
C    IF FITQ, TRY OTHER FIT ROUTINE
      IF (QSW) CALL FITE(MSR,MC,MV1,IV1,Q,KA,KB)
      RETURN
      END
C ***** FITE *****
      SUBROUTINE FITE(MSR,MC,MV1,IV1,Q,KA,KB)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      INCLUDE 'GRR.CCC'
      INCLUDE 'CONST.CCC'

      COMMON/QUE/V1,V2,VV1,VV2,G,CQXZ,CQYZ,CQX,CQY,M1,M2,P1,P2,MS,ME
1,K1,K2
      INTEGER P1,P2,ELNUM
      DIMENSION MV1( 6),IV1( 6),Q(2)
      INTEGER QUEX/6/
      DATA MASK1/ZFF000000/,MASK2/Z00FF0000/,MASK3/Z0000FF00/
      IF (.NOT.RSW) GO TO 2
      I1=SHFTR(LAND(MASK1,KA),24)-240
      I2=SHFTR(LAND(MASK2,KA),16)-240
      I3=SHFTR(LAND(MASK3,KA),8)-240
      J1=SHFTR(LAND(MASK1,KB),24)-240
      J2=SHFTR(LAND(MASK2,KB),16)-240
      J3=SHFTR(LAND(MASK3,KB),8)-240
      K1 = I1*6 +(I3-1)*2 + I2
      K2= J1*6 +(J3-1)*2 + J2
      GO TO 4
2     K1=KA
      K2=KB
4     MS = MSR
      ME = MC
      M1 = ELNUM(MV1(1))
      M2 = ELNUM(MV1(2))
      P1 = IV1(1)
      P2=IV1(2)
      MN = INFF(1,M1)
      V1 = FDAT(M1,P1)
      V2 = FDAT(M2,P2)
      VV1 = V1
      VV2 = V2
      IF (.NOT.QSW) GO TO 3
      CQXZ =DCOS(PI2 * Q(1))
      CQYZ =DCOS(PI2 * Q(2))
      GO TO 5
3     CQXZ = Q(1)/100.D0
      CQYZ = Q(2)/100.D0
5     CALL GRT(1,V1,7,QUEX)
      IF(IERR) GO TO 7
6     WRITE(3,1) P1,MV1(1),V1,P2,MV1(2),V2
1     FORMAT(//,39H PARAMETER REPLACEMENTS MADE BY FITTING,/,5X,I5,1X,

```

```

1 4H OF ,A5,3H = ,F20.12,10X,15,1X,4H OF ,A5,3H = ,F20.12)
  RETURN
7  V1 = VV1
  V2 = VV2
  CALL REPFLT(M1,P1,V1)
  CALL REPFLT(M2,P2,V2)
  GO TO 6
  END
C ***** FLDAT *****
  FUNCTION FLDAT(M, J)
    IMPLICIT REAL*8 (A-H,O-Z)
C  DATA RETRIEVED MUST BE FLOATING POINT.  SYMBOLIC NOT ALLOWED.
C  RETRIEVES ONE FLOATING POINT NUMBER FROM INDEX J OF LQ OF ELEMENT M
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'BWORK.CCC'
    LOC = INFF(10,M) + J - 1
25  FLDAT = STORE(LOC)
    RETURN
    END
C ***** FLQ *****
  SUBROUTINE FLQ(M,L,J,F)
    IMPLICIT REAL*8 (A-H,O-Z)
C  FETCH F FROM INDEX OF LQ STORAGE OF ELEMENT M
C  L=1 FOR LQ      L=2 FOR LQ2      L=3 FOR LQ3
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    LM = 10
    IF (L.EQ.2) LM = 19
    IF (L.EQ.3) LM = 24
18  LOC = INFF(LM,M) + J - 1
C  IF LQ3 OR LQ2 HAVE NOT BEEN SET, USE LQ2 OR LQ
    IF (LOC.NE.0) GO TO 15
    IF (LM.EQ.10) GO TO 15
    IF (LM.EQ.24) LM = 19
    IF (LM.EQ.19) LM = 10
    GO TO 18
15  F = STORE(LOC)
    RETURN
    END
C ***** FTIME *****
  FUNCTION FTIME(BIDON)
CC      GIVES THE ELAPSED JOB TIME IN FLOATING-POINT MINUTES
CC      BY CALLING SOME INSTALLATION-DEPENDENT SUBROUTINE
CC
    COMMON /CLOCK1/TSTART
    FTIME=SECNDS(TSTART)/60
    RETURN
    END
C ***** FXPT *****
  SUBROUTINE FXPT(M)
    IMPLICIT REAL*8 (A-H,O-Z)
C  WRITES INTEGER WORD IA INTO INFF(J,M)
C  EQUILIBRIUM ORBITS UNDER LINEAR AND/OR ARBITRARY EXTERNAL TRANSFOR-
C  MATIONS, LINEARIZED 4X4 MATRIX NEAR FIXED POINT (M4), EIGENVALUES

```

```

C  AND EIGENVECTORS OF M4, ETC.
C
C  S  FXPT KA KB//P BLIN/NSTR ISAV/ITERS IFLAG/ITAPE
C      /E1      E2      D1      D2      D3

C      KA = OUTPUT OPTION ——
C      KA = 0  MATRIX,EIGENVALUES,CLOSED ORBIT,ETA-ORBIT AT POSITION 0
C      KA = 1  OPTION 0 AND EIGENVECTORS AT POSITION 0
C      KA = 2  OPTION 1 AND TRACK VCO, ETA
C      KA = 3  OPTION 1 AND TRACK EIGENVECTORS
C      KA = 4  OPTIONS 2 AND 3 TOGETHER
C      IF KA IS NEGATIVE, PRINT OF ORBITS IS SUPPRESSED
C      AT ALL POSITIONS EXCEPT THOSE OF ELEMENTS WHOSE
C      FIRST CHARACTER IS  ( 4-8 )

C      KB = NUMBER OF SUPERPERIODS
C      P  = PREV. DEF. PART. WHOSE INPUT CONTAINS INITIAL REF. RAY
C      X  DX  Y  DY  DS  DP/P  1.
C      BLIN = BEAM LINE DEFINED BY BML INSTRUCTION
C      IF KB IS NEGATIVE, THEN
C      THE REFLECTION OF BLIN IS ADDED TO BLIN TO FORM COMPLETE SP.
C      THIS FEATURE WILL NOT WORK IF NSTR .GT. 1

C      NSTR = NUMBER OF SUPERPERIODS FOR FIXED POINT ORBIT TO CLOSE
C      ISAV = 0,1 —— DONT SAVE, SAVE CALCULATED EQ.ORBIT IN P
C      ITERS = MAXIMUM NUMBER OF ITERATIONS
C      IFLAG = 0  CLOSED ORBIT RAY ONLY
C      IFLAG = 1  PLUS 9 NEIGHBORING RAYS.
C      IFLAG = 2  PLUS 4 RAYS OF LINEARIZED EQUATIONS.

C      IFLAG = 3  PLUS 4 RAYS OF LINEARIZED EQUATIONS,
C      FIRST FOR THE ENVELOPE AND SECOND FOR THE IONS OF A BEAM.
C      ITAPE = 0,1 DO NOT, DO WRITE ETA _CLOSED ORBITS ON TAPE 12

C      E1, E2 = CONVERGENCE CRITERIA FOR DISPLACEMENTS, SLOPES
C      D1,D2 = DISP. FROM REF. RAY OF NEIGHBOR RAYS FOR
C      CALCULATING M4.
C      D3 = DISP IN DP/P FOR CALCULATING LINEARIZED 3X0 MATRICES
C      OUT = OUTPUT OPTION IN BCD

C  IF D1=0, NEIGHBORING RAYS NOT CALCULATED, LINEARIZATION WITH REVMAT.
C  IF OUT IS BLANK, EIGENVECTORS NOT CALCULATED, NO TRACKING.
C  IF OUT = 4HETRK, EIGENVECTOR ORBITS ARE TRACKED.
C  LINEARIZED 7X7 MATRIX STORED AT LQ+KADD OF FXPT INSTRUCTION
C  LINEARIZED 4X4 MATRIX STORED AT LQ+KADD+53
C  EIGENVALUES STORED AT AT LQ3, EIGENVECTORS AT LQ3+8
C  LQ2 = LQ+KADD+49+16, LQ3 = LQ2+7, KADD = 3. KIND IS 2HR7

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      INCLUDE 'BMI1.CCC'
      INCLUDE 'CONTRL.CCC'

```

```

INCLUDE 'DIM.CCC'
INCLUDE 'CORB.CCC'
COMMON/seed/nsr
DIMENSION MI(16000)
EQUIVALENCE (MI,MI1),(NELS,NELS1)
COMMON/MH/ML(2)
COMPLEX*16 CTR,CSQ,LAM,BLAM,CABS,CSQRT,CM,X,EX1,EX3
DIMENSION RW(3)
DIMENSION P(5),VW(112),V(7,16),E(4),F(7),VV(7),
1      RM(4,4),RMI(4,5),LAM(4),BLAM(4),C(4),CM(3,3),X(4,4),
2      VZ(7,7),RX(2,3),LABELA(4),LABELB(4),MU(4)
REAL*8 LBL,MU,LABELA,LABELB
DIMENSION LBL(6),Q(4),R7(7,7),VP(7),EMOD(4,2),EPI(4,2)
EQUIVALENCE (V(1),VW(1)),(VV(1),VW(78)),(VZ(1,1),VW(85))
LOGICAL ITERSW
INTEGER TRKR,R5J
DATA RW /3*0.0/
DATA NV/1,2,3,4,5,6*0/
DATA LABELA /'0X...',1H,'0Y...',1H /
DATA LABELB /'LMD1','1/LMD1','LMD3','1/LMD3'/
DATA LBL /'REAL(X1)','IMAG(X1)','REAL(X3)','IMAG(X3)',
2      'EQ ORBIT','ET ORBIT'/
DATA IBLNK/1H /
INCLUDE 'CDERIV.CCC'

```

C

```

nnn=nsr
IIH = 1
IIV = 1

VP(5)=0.
VP(6)=1.
VP(7)=0.
TPI = 4.D0*DARCOS(0.D0)
KA = INFF(4,M)
KA = IABS(KA)
KB = INFF(5,M)
SPER = DFLOAT(IABS(KB))
NSTR = IDAT(M,1)
IF (NSTR.EQ.0) NSTR=1
IVSV = IDAT(M,2)
ITERS=IDAT(M,3)
IFLAG = IDAT(M,4)
ITAPE = IDAT(M,5)

```

C RETRIEVE FLOATING POINT INPUT E1,E2,D1,D2

```

CALL DATA(M,1,1,5,P)
EPSW = P(1)
EPST = P(2)
DELP = P(3)
DELT = P(4)
DELP = P(5)
IF (EPSW.EQ.0.) EPSW=1.D-08
IF (EPST.EQ.0.) EPST=1.D-08
IF (DELP.EQ.0) DELP = 1.D-08
NGO = 2
IF (IFLAG.EQ.0) GO TO 2

```

```

      NGO=1
      IF (DELW.EQ.0.) DELW=1.D-08
      IF (DELT.EQ.0.) DELT=1.D-08
2     ML(1)=M7END-3
      ML(2)=M7END-4
C    SET UP BEAM SYSTEM ARRAY MI
      MBML = MDAT(M,2)
      NDAT = INFF(17,MBML)
      CALL MIFILL(MBML,1,NDAT,NELS,MI)
      IF (ERROR) RETURN
C    OPTIONS ON ITERATIONS AND PRINT
C    SET UP ARRAY FOR PREVIOUSLY DEFINED PARTICLE
C    SET UP INITIAL REFERENCE RAY
C    ZERO VW ARRAY
      DO 3 I=1,112
      VW(I) = 0.0
3     CONTINUE
      NP = MDAT(M,1)
      CALL DATA (NP,1,1,7,VW)
      NV(1) = 1
33    MB=1
      ME = NELS
      NITS = 0
      IF( ITERS.EQ.0 ) ITERS = 25
      ITERSW = .FALSE.
C
      E(1) = EPSW
      E(2) = EPST
      E(3) = EPSW
      E(4) = EPST
      F(1) = DELW
      F(2) = DELT
      F(3) = DELW
      F(4) = DELT
      CALL HED
      NPM = INFF(2,NP)
      NAM = INFF(2,M)
      WRITE(3,1018)
      WRITE(3,1000) NAM,NPM,(VW(I),I=1,7)
      MM = 1
      IOP = IFLAG
      GO TO (31,32), NGO
32    MN = 1
391   CALL RTRV7(MUNIT,R7,RW)
      CALL STOR7(M,R7,RW)
      GO TO 4
31    MN=9
      MR = 4
      NR = 1
      ASSIGN 4 TO NRS
      IF (IFLAG.EQ.1) GO TO 800
      NR=5
      MN=1
      GO TO 802
39    IF (IFLAG.NE.1) GO TO 802

```



```

      GO TO 800
C  SAVE INITIAL CONDITIONS OF THE REFERENCE RAY.
  4   DO 5 I=1,7
      5 VV(I) = VW(I)
      IF (IFLAG.EQ.4) CALL DATA(M,7,1,7,V(1,8))
C  TRACK PARTICLES MM TO MN THROUGH BEAMLINE ELEMENTS MB TO ME.
      ASSIGN 510 TO TRKR
      DO 1 ISTR=1,NSTR
        nsr=nrn
        GO TO 900
  510 CONTINUE
  1   CONTINUE
      GO TO (51,52), NGO
C CALCULATE THE 4X4 MATRIX, M.
  51  IF (IFLAG.EQ.1) GO TO 511
      DO 512 J=1,4
      DO 512 I=1,4
  512  RM(I,J) = V(I,J+1)
      GO TO 513
  511  DO 55 J = 1 , 4
      N = J + 1
      DN=.5D0/F(J)
      DO 55 I=1,4
  55   RM(I,J) = ( V(I,N) - V(I,N+4) ) * DN
  513  IF (KB.GE.0) GO TO 56
      RX(1,1) = RM(2,1)
      RX(2,1) = RM(4,1)
      RX(1,2) = RM(2,3)
      RX(2,2) = RM(4,3)
      RX(1,3) = -V(2,1)
      RX(2,3) = -V(4,1)
      GO TO 56
  52   CALL RTRV7(M,R7,RW)
      IF (KB.GE.0) GO TO 57
      RX(1,1)=R7(2,1)
      RX(2,1)=R7(4,1)
      RX(1,2)=R7(2,3)
      RX(2,2)=R7(4,3)
      RX(1,3)=- V(2,1)
      RX(2,3)= -V(4,1)
  57   DO 53 J=1,4
      DO 53 I=1,4
  53   RM(I,J) = R7(I,J)
  56   CONTINUE
      IF (KB.GE.0) GO TO 58
      CALL MXDIV(RX,2,1,D,2)
      V(1,1)=RX(1,3)+VV(1)
      V(3,1)=RX(2,3)+VV(3)
      DX=V(2,1)
      DY=V(4,1)
      V(2,1)=0.
      V(4,1)=0.
      NITS = NITS + 1
      IF ( (DABS(DX).LT.E(2)).AND.(DABS(DY).LT.E(2)) ) GO TO 146
      IF (NITS.GT.ITER) GO TO 146

```

```

      GO TO (39,391), NGO
146  IF (NITS.GT.ITER) ITERSW = .TRUE.
      GO TO (141,142), NGO
58   CONTINUE
C   SET UP N - I AND REF RAY JUMP (RHS).
      DO 7 I = 1, 4
      DO 6 J = 1, 4
        6 RMI(I,J) = RM(I,J)
        7 RMI(I,I) = RMI(I,I) - 1.0D0
      DO 61 I=1,4
61   RMI(I,5) = VV(I)-V(I,1)
C   SOLVE FOR EQ.-ORBIT-INITIAL CONDITIONS ( IN TERMS OF REF. RAY ).
      CALL MXDIV( RMI, 4, 1, D, 4 )
      IF( D.NE.0. ) GO TO 75
      WRITE( 3, 1006 )
      WRITE (3,1001) ((RM(I,J),J=1,4),I=1,4)
1001 FORMAT (/10X,4F16.8)
      CALL HED
      RETURN
C   CALCULATE EQ.-ORBIT-INITIAL-CONDITIONS.
75   DO 8 I=1,4
8     V(I,1) = RMI(I,5) + VV(I)
C   CHECK FOR CONVERGENCE.
      NITS = NITS + 1
      IF (KA.LT.2) GO TO 14
      WRITE (3,1013) NITS,(V(I,1),I=1,4)
14   CONTINUE
      DO 9 I = 1, 4
      IF( DABS( RMI(I,5) ).LT.E(I) ) GO TO 9
      IF (NITS.GT.ITER) GO TO 9
      GO TO (39,391), NGO
9     CONTINUE
      GO TO (141,142), NGO
141  CALL RTRV7(MUNIT,R7,RW)
      IOP = 1
      DO 200 J=1,4
      DO 200 I=1,4
200  R7(I,J)=RM(I,J)
      MM = 10
      MN = 11
      NR=1
      IF (IFLAG.EQ.1) GO TO 138
      IF (IFLAG.EQ.4) GO TO 142
      MM = 6
      MN = 7
138  DO 143 I=1,7
      V(I,MM) = VW(I)
143  V(I,MN) = VW(I)
      V(6,MM) = VW(6) + DELP
      V(6,MN) = VW(6) - DELP
      ASSIGN 144 TO TRKR
      GO TO 900
144  DO 145 I=1,5
145  R7(I,6) = 0.5D0*(V(I,MM)-V(I,MN))/DELP
      ASSIGN 142 TO R5J

```

```

      GO TO 805
142  IOP = IFLAG
      CALL ROW5(R7)
      IF (KB.GE.0) GO TO 149
      ML1=ML(1)
      ML2=ML(2)
      CALL STOR7(ML1,R7,RW)
      CALL REF(ML2,ML1)
      CALL MMM(ML1,2,ML)
      CALL RTRV7(ML1,R7,RW)
      DO 139 J=1,4
      DO 139 I=1,4
139  RM(I,J) = R7(I,J)
149  IF(NGO.EQ.2) GO TO 1400
      DO 151 I=1,7
151  VV(I) = VW(I)
      CALL MXV7(R7,VV,VV)
      DO 152 I=1,5
152  R7(I,7) = VW(I) - VV(I)
1400 CONTINUE
      ML1 = ML(1)
      CALL STOR7(ML1,R7,RW)
C  CALCULATE OFF-MOMENTUM ORBIT VP AND STORE AT LQ3
140  DO 148 I=1,4
      RMI(I,5)=-R7(I,6)
      DO 147 J=1,4
147  RMI(I,J)=R7(I,J)
148  RMI(I,I)=RMI(I,I)-1.D0
      CALL MXDIV(RMI,4,1,D,4)
      DO 150 I=1,4
150  VP(I)=RMI(I,5)
      CALL STDAT(M,8,1,7,VP)
C  REPLACE INITIAL CONDITIONS OF PARTICLE WITH COMPUTED VALUES
      IF ((IVSV.NE.0).AND.(IFLAG.NE.4))
1    CALL STDAT(NP,1,1,4,VW)
C  STORE CLOSED ORBIT VECTOR AT LQ2
      VW(5)=0.
      IF (IFLAG.NE.4) CALL STDAT(M,7,1,7,VW)
      NITP = NITS - 1
      IF (ITERSW) WRITE (3,1019) NITP
      IF (.NOT.ITERSW) WRITE (3,1020) NITP
      WRITE (3,1007) NAM, ( (R7(I,J),J=1,7) , I = 1,7)
C  CALCULATE THE EIGENVALUES OF M.
      TX = RM(1,1) + RM(2,2)
      TY = RM(3,3) + RM(4,4)
      CTR = DCMPLEX( TX + TY, 0.D0 )
      CSQ = CDSQRT( DCMPLEX( ( TX - TY )**2 + 4.D0*( ( RM(1,3) + RM(4,2))
1          *( RM(2,4) + RM(3,1) ) - ( RM(1,4) - RM(3,2) )
2          *( RM(2,3) - RM(4,1) ) ), 0.D0 ) )
      BLAM(1) = 0.5D0*( CTR + CSQ )
      BLAM(2) = 0.5D0*( CTR - CSQ )
      IF( TX.GE.TY ) GO TO 41
C  SWITCH BLAM(1) AND BLAM(2) IF TX.LT.TY
      LAM(1) = BLAM(1)
      BLAM(1) = BLAM(2)

```

```

      BLAM(2) = LAM(1)
41  LAM(1) = 0.5D0*( BLAM(1) + CDSQRT( BLAM(1)*BLAM(1) -(4.D0, 0.D0)))
      LAM(2) = 1.D0/LAM(1)
      LAM(3) = 0.5D0*( BLAM(2) + CDSQRT( BLAM(2)*BLAM(2) -(4.D0, 0.D0)))
      LAM(4) = 1.D0/LAM(3)
C  WRITE OUT THE EIGENVALUES...BOTH IN CARTESIAN AND POLAR FORMS.
      DO 42 I = 1, 4
        C(I) = CDABS(LAM(I))
        DOLREALA=DIMAG(LAM(I))
        DOLREALB=DREAL(LAM(I))
        MU(I) = DATAN2(DOLREALA,DOLREALB)
        Q(I) = SPER*MU(I)/TPI
        Q(I) = DMOD(Q(I),1.D0)
        IF (Q(I).LT.0.) Q(I)=Q(I)+1.D0
42  CONTINUE
C  STORE EIGENVALUES AT LQ3+7
      CALL STDAT(M,8,8,8,LAM)
      WRITE( 3, 1004 )
43  WRITE( 3, 1005 ) ( LABELA(I), LABELB(I), LAM(I), I, C(I), I,
1      MU(I), I, Q(I), I = 1, 4 )
      IF (KA.EQ.0) GO TO 93

C  COMPUTE EIGENVECTORS.
C  FIRST AND SECOND EIGENVECTORS.
C  FIRST COMPONENTS.
35  DO 45 L=1,3
      DO 44 J = 1, 3
        RT = RM(L+1,J+1)
44  CM(L,J) = DCMPLX(RT,0.D0)
      45 CM(L,L) = CM(L,L) - LAM(1)
      CALL DET( CM, X(1,1) )
C  SECOND THROUGH FOURTH COMPONENTS.
      D = -1.D0
      DO 50 L = 2, 4
        I = L - 1
        DO 46 J = 1, 3
          RT = RM(J+1,I)
46  CM(J,I) = DCMPLX(RT,0.D0)
          IF( L.EQ.3 ) CM(1,2) = CM(1,2) - LAM(1)
          IF( L.EQ.4 ) CM(2,3) = CM(2,3) - LAM(1)
          CALL DET( CM, X(L,1) )
          X(L,1) = D*X(L,1)
          D = -D
50  CONTINUE
      C1 = CDABS( X(1,1) )
      EX1 = X(1,1)/C1
      DO 536 I=1,4
536 X(I,1) = X(I,1)/EX1
      RHO1 = DIMAG( DCONJG(X(1,1))*X(2,1) + DCONJG(X(3,1))*X(4,1) )
      FAC = 1.D0
      IF (RHO1.NE.0.) FAC = 1.D0/DSQRT(DABS(RHO1))
      DO 537 I=1,4
        X(I,1) = FAC * X(I,1)
537 X(I,2) = DCONJG(X(I,1))
      L = 1

```

```

      ASSIGN 467 TO IWR
      GO TO 998
467 L = 2
      ASSIGN 468 TO IWR
      GO TO 998
C  THIRD AND FOURTH EIGENVECTORS.
C  FIRST COMPONENTS.
468 DO 531 I = 1, 2
      DO 531 J = 2, 4
      RT = RM(I,J)
531 CM(I,J-1) = DCMLPX(RT,0.D0)
      DO 532 J = 2, 4
      RT = RM(4,J)
532 CM(3,J-1) = DCMLPX(RT,0.D0)
      CM(2,1) = CM(2,1) - LAM(3)
      CM(3,3) = CM(3,3) - LAM(3)
      CALL DET( CM, X(1,3) )
      X(1,4) = DCONJG( X(1,3) )
C  SECOND THROUGH FOURTH COMPONENTS.
      D = -1.D0
      DO 533 L = 2, 4
      I = L - 1
      CM(1,I) = DCMLPX( RM(1,I), 0.D0 )
      CM(2,I) = DCMLPX( RM(2,I), 0.D0 )
      CM(3,I) = DCMLPX( RM(4,I), 0.D0 )
      IF(L.EQ.2 ) CM(1,1) = CM(1,1) - LAM(3)
      IF( L.EQ.3 ) CM(2,2) = CM(2,2) - LAM(3)
      CALL DET( CM, X(L,3) )
      X(L,3) = D*X(L,3)
      D = -D
533 CONTINUE
      C3 = CDABS( X(3,3) )
      EX3 = X(3,3)/C3
      DO 546 I=1,4
546 X(I,3) = X(I,3)/EX3
      RHO3 = DIMAG( DCONJG(X(1,3))*X(2,3) + DCONJG(X(3,3))*X(4,3) )
      FAC = 1.D0
      IF (RHO3.NE.0.) FAC = 1.D0/DSQRT(DABS(RHO3))
      DO 547 I=1,4
      X(I,3) = FAC * X(I,3)
547 X(I,4) = DCONJG(X(I,3))
      L = 3
      ASSIGN 534 TO IWR
      GO TO 998
534 L = 4
      ASSIGN 535 TO IWR
      GO TO 998
C  1. TRACE EQUILIBRIUM ORBIT AND EIGENVECTOR PARTS THROUGH SYSTEM,
C  2. LINEARIZE INDIVIDUAL NLTS.
535 CONTINUE
C  SET UP EIGENVECTOR INITIAL CONDITIONS.
      DO 91 I = 1, 3, 2
      DO 91 J=1,4
      VZ(J,I) = DREAL( X(J,I) )
      VZ(J,I+1) = DIMAG( X(J,I) )

```

91 CONTINUE

C STORE 4 EIGENVECTORS AT LQ2+7, 7 COMPONENTS EACH.

C ORDER— REAL X1, IMAG X1, REAL X3, IMAG X3.

DO 92 I=1,4

DO 92 J=5,7

92 VZ(J,I) = 0.

DO 97 I=5,7

DO 97 J=1,7

97 VZ(J,I) = 0.

IQ=8

DO 90 I=1,4

CALL STDAT(M,7,IQ,7,VZ(1,I))

IQ=IQ+7

90 CONTINUE

C WRITE CLOSED AND OFF-MOMENTUM ORBITS AT POSITION 0

93 WRITE(3,1017)

WRITE(3,1014) LBL(5),(V(I,1),I=1,7)

WRITE(3,1014) LBL(6),(VP(I),I=1,7)

IF (KA.EQ.0) GO TO 94

WRITE(3, 1002)

C WRITE INITIAL CONDITIONS.

L=0

MATL = IBLNK

ASSIGN 600 TO IBET

GO TO 599

600 IF (IFLAG.EQ.0) CALL STOR7(M,VZ,RW)

IF (IFLAG.EQ.4) CALL DATA(M,7,1,7,V(1,8))

94 II = 0

nsr=nrn

100 II = II + 1

L = L + 1

MB = II

ME = II

C SAVE INITIAL CONDITIONS.

DO 101 J=1,7

101 VV(J) = VW(J)

IF (IFLAG.NE.4) GO TO 103

DO 104 J=1,7

104 VP(J) = V(J,8)

103 ASSIGN 102 TO TRKR

MM = 1

MN = 1

NR=1

GO TO 900

102 CONTINUE

IF (KA.LT.3.OR.IFLAG.NE.0) GO TO 604

CALL RTRV7(M,VZ,RW)

ASSIGN 604 TO IBET

GO TO 599

604 CONTINUE

GO TO (95,40), NGO

95 IF(NL.EQ.0) GO TO 40

```

C  LINEARIZE NON-LIN. TRANS
C  FORM 3X3 MATRICES
C  RESTORE INITIAL CONDITIONS.
      DO 111 J=1,7
      F(J) = VW(J)
111  VW(J) = VV(J)
      IF (IFLAG.NE.4) GO TO 105
      DO 106 J=1,7
106  V(J,8) = VP(J)
105  ASSIGN 112 TO NRS
      ASSIGN 116 TO TRKR
      MM=1
      MN=11
      JP=10
      JQ=11
      IF (IFLAG.EQ.1) GO TO 800
      MN=1
      JP=6
      JQ=7
      NR=5
      IF (IFLAG.NE.4) ASSIGN 114 TO TRKR
      GO TO 802
112  DO 113 I=1,7
      V(I,JP) = VW(I)
113  V(I,JQ) = VW(I)
      V(6,JP) = VW(6) + DELP
      V(6,JQ) = VW(6) - DELP
      GO TO 900
114  MM=6
      MN=7
      NR=1
      IOP = 1
      ASSIGN 115 TO TRKR
      GO TO 900
115  IOP = IFLAG
C  PUT LINEARIZED 3X3 MATRIX REPRESENTING AN EXT IN ITS STORAGE
116  CALL RTRV7(MUNIT,R7,RW)
      IF (IFLAG.EQ.1) GO TO 15
      DO 16 J=1,4
      DO 16 I=1,4
16   R7(I,J) = V(I,J+1)
      GO TO (15,17,17,18), IFLAG
15   D=DELW
      DO 17 J=1,4
      D=DELW*2.D0
      IF (J.EQ.2.OR.J.EQ.4) D=DELT*2.D0
      DO 11 I=1,4
11   R7(I,J) = ( V(I,J+1) - V(I,J+5) ) / D
17   DO 12 I=1,4
12   R7(I,6) = 0.5D0 * ( V(I,JP) - V(I,JQ) ) / DELP
      ASSIGN 18 TO R5J
      GO TO 805
18   CALL MXV7(R7,VV,VV)
      DO 13 I=1,5
13   R7(I,7) = VW(I) - VV(I)

```

```

      DO 19 I=1,7
19   VW(I) = F(I)
      RW(1)=FDAT(NL,1)
      CALL STOR7(NL,R7,RW)
40   IF (II.LT.NELS) GO TO 100

402  CONTINUE

C   TRACK BETA FUNCTIONS, ETA AND CLOSED ORBITS THROUGH BEAM SYSTEM.
C   TRACKING DONE IN CYX, WHICH CALLS TRKCN FOR ORBIT TRACKING.
C   BETA FUNCTIONS FORMED FROM 2X2 SUBMATRICES ARE ONLY APPROXIMATE
C   IF THERE IS COUPLING.

C   TEST TRACE
      TRX = RM(1,1) + RM(2,2)
      TRY = RM(3,3) + RM(4,4)
      IF ( (DABS(TRX).LT.2.) .AND. (DABS(TRY).LT.2.) ) GO TO 405
      WRITE (3,1016)
1016 FORMAT (5X,'X OR Y BETATRON MOTION UNSTABLE.')
405  CONTINUE
      IF (KB.GE.0) GO TO 406
      NELS=NELS+1
      MI(NELS) = M7END - 4
406  CONTINUE
      ML1 =ML(1)
      CALL RTRV7(ML1,R7,RW)
      CALL STOR7(M,R7,RW)
      IF (IFLAG.NE.4) GO TO 409
      CALL DATA(M,7,1,7,V(1,8))
      CALL STDAT(M,7,1,7,VW)
409      nsr=nrn
      CALL CYX(M)
      nsr=nrn
      CALL HED
      IF (IFLAG.NE.3) RETURN
      IF (KB.LT.0) NELS=NELS-1
      DO 408 I=1,6
408  VW(I) = 0.
      IFLAG = 4
      NV(1)=8
      GO TO 33

C   SET UP THE EIGHT NEIGHBORING RAYS
800  DO 801 J=8,63
801  VW(J)=VW(J-7)
      V(1,6)=V(1,6)-DELW
      V(2,7)=V(2,7)-DELT
      V(3,8)=V(3,8)-DELW
      V(4,9)=V(4,9)-DELT
      V(1,2) = V(1,2) + DELW
      V(2,3) = V(2,3) + DELT
      V(3,4) = V(3,4) + DELW
      V(4,5) = V(4,5) + DELT
      GO TO NRS, ( 4, 112 )
802  DO 803 J=8,35

```



```

803 VW(J) = 0.
DO 804 I=1,4
804 V(I,I+1) = 1.D0
CALL STOR7(M,V(1,2),RW)
GO TO NRS,(4,112)
905 DO 806 J=1,4
806 R7(5,J) = R7(2,J) * R7(1,6) - R7(1,J) * R7(2,6)
1      + R7(4,J) * R7(3,6) - R7(3,J) * R7(4,6)
GO TO R5J,(142,18)
C TRACK PARTICLES MM THROUGH MN THROUGH BEAM ELEMENTS MB THROUGH ME.
900 NLF = 0
DO 950 J = MB, ME
NL = MI(J)
MATL = INFF(2,NL)
DO 950 MT = MM,MN
CALL TRKCN( M, NL, V(1,MT), NP, IOP )
KIND = INFF(20,NL)
IF (KIND.EQ.3HPRD.OR.KIND.EQ.3HMAP) NLF = NL
950 CONTINUE
NL = NLF
GO TO TRKR,(102,114,115,116,144,510)
C WRITE OUT AN EIGENVECTOR.
998 IF (KA.NE.0) WRITE (3,1008) LAM(L),(X(J,L),J=1,4)
GO TO IWR, ( 467, 468, 534, 535 )
C
C PRINT EIGENFUNCTIONS IN POLAR FORM
599 CONTINUE
DO 601 K=1,2
DO 601 I=1,4
J = 2 * K - 1
IF ( (VZ(I,J).NE.0.) .OR. (VZ(I,J+1).NE.0.) ) GO TO 602
EMOD(I,K) = DCMLPX(0.D0,0.D0)
EPHI(I,K) = DCMLPX(0.D0,0.D0)
GO TO 601
602 EMOD(I,K) = DSQRT ( VZ(I,J)*VZ(I,J) + VZ(I,J+1)*VZ(I,J+1) )
EPHI(I,K) = DATAN2 ( VZ(I,J+1),VZ(I,J) )
601 CONTINUE
WRITE (3,1003) L,MATL, ( EMOD(I,1),EPHI(I,1),I=1,4 ) ,
1      ( EMOD(I,2),EPHI(I,2),I=1,4 )
GO TO IBET,(600,604)

1000 FORMAT(64H0CALCULATION OF THE EQUILIBRIUM ORBIT AND BETATRON FUNCT
1IONS OF A5,1H./34H INITIAL REFERENCE RAY DEFINED BY A5/7H X = ,
2 F11.8,3X,'DX = ',F11.8,3X,'Y = ',F11.8,3X,'DY = ',F11.8,
3 5X,'DS = ',F11.8,3X,'DP/P = ',F11.8,3X,F11.8)
1002 FORMAT ( 1H1,'EIGENVECTORS 1 AND 3 IN POLAR COORDINATES'/
1 1X'POS'23X'X1 '22X'DX1'22X'Y1 '22X'DY1'/
2 27X'X3 '22X'DX3'22X'Y3 '22X'DY3'/ )
1003 FORMAT ( 1X,I4,'X,A5,1X, 4(2F11.6,3X) / 12X, 4(2F11.6,3X) / )
1004 FORMAT ('0EIGENVALUES OF THE 4X4 SUBMATRIX')
1005 FORMAT(2A6,4H = (,2F12.8,5X,6H ), C(I1,4H) = F12.8,5H, MU(I1,
1 4H) = F12.8,12H RAD, Q(I1,4H) = F12.8 )
1006 FORMAT(21H0SINGULAR 4X4 MATRIX.)
1007 FORMAT ('07X7 MATRIX FOR ',A5/(10X,7F16.8))
1008 FORMAT(16H0EIGENVALUE = ( F16.8,2H, F16.8,21H ), EIGENVECTOR = (

```

```

1,F16.8,2H, F16.8,2H )/(69X,2H( F16.8,2H, F16.8,2H )))
1013 FORMAT(12H ITERATION =,I4,4H XO=,F16.8,5H DXO=,F16.8,4H YO=,F16.8,
15H DYO=,F16.8)
1014 FORMAT (/1X,A10,7F15.8)
1018 FORMAT ( 1H1)
1017 FORMAT(/15X,7X,' X',13X,'DX',13X,' Y',13X,'DY',13X,'DS',13X,'DP/P'
1 )
1019 FORMAT (2X,'CONVERGENCE CRITERIA NOT REACHED AFTER MAXIMUM '
1 'ITERATIONS OF ' I4 '.')
1020 FORMAT (1X,'CLOSED ORBIT FOUND AFTER ' I4' ITERATIONS.')
```

C

END

C ***** GENNUM *****

SUBROUTINE GENNUM (NM,KC,LIST)

IMPLICIT REAL*8 (A-H,O-Z)

C GENERATE NUMBERED NAMES. E.G., N1,N2,N3...NKC IN BCD WHERE

C N IS FIRST LETTER OF NM. PUT NAMES INTO ARRAY LIST.

DIMENSION LIST(1)

C STRIP OFF FIRST LETTER OF NM

INTEGER MSK1/ZFF000000/ ,FILL/Z00004040/

IXdol=LAND(NM,MSK1)

IJ=1

K1=240

K2=241

K3=241

DO 100 JA=1,KC

K1=K1+1

GO TO (25 ,30 ,35),IJ

25 LIST(JA)=IXdol+65536*K1+FILL

IF (K1.NE.249) GO TO 100

26 IJ=IJ+1

27 K1=239

GO TO 100

30 LIST(JA)=IXdol+65536*K2+256*K1+64

IF (K1.NE.249) GO TO 100

K2=K2+1

IF (K2.NE.250) GO TO 27

K2=240

GO TO 26

35 LIST(JA)=IXdol+65536*K3+256*K2+K1

IF (K1.NE.249) GO TO 100

K2=K2+1

IF (K2.NE.250) GO TO 27

K2=240

K3=K3+1

IF (K3 .EQ. 250) GO TO 105

GO TO 27

100 CONTINUE

RETURN

105 WRITE(3,5)

5 FORMAT (5X,' CANNOT GENERATE NUMBERS GREATER THAN 999.')

RETURN

END

C ***** GRT *****

```

SUBROUTINE GRT( N, C, IN, AUX )
IMPLICIT REAL*8 (A-H,O-Z)
INTEGER AUX
INCLUDE 'CONTRL.CCC'
INCLUDE 'GRR.CCC'

LOGICAL IND,INF
DIMENSION C(5)
DATA FC /1.0D-10/
DO 100 L=1,N
  IERR=.FALSE.
  JK=0
  IF (C(L))45,46,45
45  RT=.9D0*C(L)
  ASSIGN 1 TO NN
  GO TO 80
1   X0=FPRT
  RT=1.1D0*C(L)
  ASSIGN 2 TO NN
  GO TO 80
2   X1=FPRT
  RT=C(L)
  ASSIGN 3 TO NN
  GO TO 80
3   X2=FPRT
  H=-.1D0*C(L)
  GO TO 50
46  RT=-1.D0
  ASSIGN 4 TO NN
  GO TO 80
4   X0=FPRT
  RT=1.
  ASSIGN 5 TO NN
  GO TO 80
5   X1=FPRT
  RT=0.
  ASSIGN 6 TO NN
  GO TO 80
6   X2=FPRT
  H=-1.D0
50  D=-.5D0
49  DD=1.+D
  BI=(X0*D*D)-(X1*DD*DD)+(X2*(DD+D))
  DEN=BI*BI-(4.D0*X2*D*DD)*(X0*D-(X1*DD)+X2)
  IF( DEN ) 52,52,51
52  IF (IN)36,35,36
35  WRITE(3,25) L,DEN
36  DEN=0.
51  DEN=DSQRT(DEN)
53  DN=BI+DEN
  DM=BI-DEN
  IF (DABS(DN)-DABS(DM)) 57,57,56

```

```

56  DEN=DN
    GO TO 58
57  DEN=DM
58  IF (DEN)55,54,55
54  DEN=1.
55  DI=(-2.*X2*DD)/DEN
    H=DI*H
    RT=RT+H
    IF (DABS(H/RT)-1.0D-8) 75,75,60
60  ASSIGN 7 TO NN
    GO TO 80
    7 IF (DABS(FPRT)-DABS(X2*10.D0)) 62,61,61
61  DI=DI*.5
    H=H*.5
    RT=RT-H
    GO TO 80
62  X0=X1
    X1=X2
    X2=FPRT
    D=DI
    GO TO 49
75  ASSIGN 751 TO IRET
750 GO TO (201,208,208,208,208,206,207,208),AUX
201 CALL AUXC(RT,FRT)
    GO TO IRET,(751,861)
206 CALL QUEX(RT,FRT)
    GO TO IRET,(751,861)
207 CALL QUEY(RT,FRT)
208 GO TO IRET;(751,861)
751 IF (ERROR) RETURN
    IF(JK.LT.150) GO TO 76
    IERR=.TRUE.
74  WRITE(3,28) RT
28  FORMAT (55H GRT UNABLE TO FIND ROOT IN 150 ITERATIONS.LAST ROOT IS
    1, E20.8)
76  IND = .FALSE.
    INF = .FALSE.
    IF(IND.OR.INF) IERR=.TRUE.
78  IF(IERR) WRITE(3,29)
29  FORMAT (32H ROOT IS INDEFINITE OR INFINITE.//)
    C(L)=RT
    IF (IN)100,77,100
77  WRITE(3,26) L,RT,FRT
100 CONTINUE
33  RETURN
80  JK=JK+1
    IF(150-JK) 75,75,86
86  ASSIGN 861 TO IRET
    GO TO 750
861 IF (ERROR) RETURN
    IND = .FALSE.
    INF = .FALSE.
    IF(IND.OR.INF) IERR=.TRUE.
    IF(IERR) GO TO 78
    FPRT=FPRT

```

```

      IF (L-1)81,91,81
81   DO 82 I=2,L
      TEM=RT-C(I-1)
      IF (DABS(TEM)-FC) 85,82,82
82   FPRT=FPRT/TEM
91   IF (IN) 89,83,89
83   WRITE(3,25) L,RT,FRT,FPRT
      89 IF (DABS(FRT)-FC) 90,84,84
      90 IF (DABS(FPRT)-FC) 76,84,84
84   GO TO NN,(1,2,3,4,5,6,7)
85   RT=RT+.001D0
      IF (IN) 88,87,88
87   WRITE(3,25) L
88   GO TO 80
25   FORMAT (I3,3E20.8)
26   FORMAT (I3,2E20.8/1H )
27   FORMAT (//)
      END

```

```

C ***** HED *****
      SUBROUTINE HED
C      PRINT A LINE OF DOTS.
      WRITE(3,1000)
1000  FORMAT (1X,130(1H.))
      RETURN
      END

```

```

C ***** HESSE *****
      SUBROUTINE HESSE
      IMPLICIT REAL*8 (A-H,O-Z)
CC      CALCULATES THE FULL SECOND-DERIVATIVE MATRIX OF FCN
CC      BY TAKING FINITE DIFFERENCES.  INCLUDES SOME SAFEGUARDS
CC      AGAINST NON-POSITIVE-DEFINITE MATRICES, AND IT MAY SET
CC      OFF-DIAGONAL ELEMENTS TO ZERO IN ATTEMPT TO FORCE POSITIVENESS
CC

```

```

      DIMENSION YY(15),GY(30)
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON /DERIVA/ G(30) ,G2(30)
      DATA DFWANT,DFZERO,DFMIN,DFMAX/.01D0,.00000001D0,.001D0,.1D0/
      WRITE (ISYSWR, 500)
      IFLAG = 4
      NPFN = NFCN
      NPARD = NPAR

```

```

C      . . . . . DIAGONAL ELEMENTS .
      MDIAG = 0
      DO 100 ID= 1, NPARD
      I = ID + NPAR - NPARD
      D = 0.02D0* DABS(DIRIN(I))
      IF (ISW(2) .GE. 1) D = 0.02D0* DSQRT(DABS(V(I,I))*UP)
      IF (D .LT. 1.0D-8 *DABS(X(I))) D = 1.0D-8 * DABS(X(I))
      DO 20 J= 1, NPAR

```

```

20 V(I,J) = 0.D0
   ICYC = 0
40 DIRIN(1) = D
   XTF = X(I)
   X(I) = XTF + D
   CALL INTOEX(X)
   CALL FCN(NPAR, GY, FS1, U, IFLAG)
   NFCN = NFCN + 1
   X(I) = XTF - D
   CALL INTOEX(X)
   CALL FCN (NPAR, GY, FS2, U, IFLAG)
   NFCN = NFCN + 1
   X(I) = XTF

```

C CHECK IF STEP SIZES APPROPRIATE

```

   ICYC = ICYC + 1
   IF (ICYC .GE. 4) GO TO 55
   DF = DMAX1(DABS(FS1-AMIN),DABS(FS2-AMIN))/UP
   IF (DF .GT. DFMIN) GO TO 45
   IF (DF .GT. DFZERO) GO TO 50
   D = D*1000.D0
   GO TO 40
45 IF (DF .LT. DFMAX) GO TO 55
50 CHAN = DSQRT(DFWANT/DF)
   IF (CHAN .LT. 0.001D0) CHAN = 0.001D0
   D = D*CHAN
   GO TO 40
55 CONTINUE

```

C GET FIRST AND SECOND DERIVATIVE

```

   G(1) = (FS1-FS2)/(2.0D0 * D)
   G2(1) = (FS1 + FS2 - 2.0D0*AMIN) / D**2
   YY(1) = FS1
   IF (DABS(G(1))+ DABS(G2(1)).GT. 1.0D-30) GO TO 80

```

C FIX A PARAMETER IF G = G2 = 0.0 .

```

   IF (ITAU .GE. 1) GO TO 85
   ISW(2) = 0
   CALL FIXPAR(I, 1, IFIX)
   WRITE (ISYSWR, 460) IFIX, NAM(IFIX), G(1), G2(1)
   IF (NPAR .EQ. 0) MDIAG = 1
   GO TO 100
80 IF (G2(1) .GT. 1.0D-30) GO TO 90
85 MDIAG = 1
   WRITE (ISYSWR, 510) I
90 V(I,I) = G2(1)
100 CONTINUE

```

```

   CALL INTOEX(X)
   IF (MDIAG .EQ. 1) GO TO 390
   ISW(2) = 1

```

C OFF-DIAGONAL ELEMENTS

```

   IF (NPAR .EQ. 1) GO TO 214
   NPARM1 = NPAR - 1
   DO 200 I= 1, NPARM1
   IP1 = I + 1
   DO 180 J= IP1, NPAR
   IF (NFCNMX-NFCN+MPFN .LT. NPAR) GO TO 210
   XTI = X(I)

```

```

      XTJ = X(J)
      X(I) = XTI + DIRIN(I)
      X(J) = XTJ + DIRIN(J)
      CALL INTOEX(X)
      CALL FCN(NPAR, GY, FS1, U, IFLAG)
      NFCN = NFCN + 1
      X(I) = XTI
      X(J) = XTJ
      ELEM = (FS1+AMIN-YY(I)-YY(J)) / (DIRIN(I)*DIRIN(J))
      IF (ELEM**2 .LT. G2(I)*G2(J)) GO TO 170
      ELEM = 0.D0
      WRITE (ISYSWR, 470) I,J
170  V(I,J) = ELEM
      V(J,I) = ELEM
180  CONTINUE
200  CONTINUE
      GO TO 214
210  J = J - 1
      WRITE (ISYSWR, 490) I,J
214  CALL INTOEX(X)
      CALL VERMIN(V,MAXINT,MAXINT,NPAR,IFAIL)
      IF (IFAIL .LT. 1) GO TO 222
      WRITE (ISYSWR,520)
C
      . . . . . DIAGONAL MATRIX ONLY . . . .
216  WRITE (ISYSWR,540)
      ISW(2) = 1
      DO 220 I= 1, NPAR
      DO 218 J= 1, NPAR
218  V(I,J) = 0.D0
220  V(I,I) = 1.0D0/G2(I)
      MDIAG = 0
      GO TO 223
222  WRITE (ISYSWR, 480)
      ISW(2) = 2
C
      . . . . . CALCULATE E D M
223  DO 225 I= 1, NPAR
      DO 225 J= 1, NPAR
225  V(I,J) = 2.0D0 * V(I,J)
      SIGMA = 0.D0
      DO 250 I= 1, NPAR
      IF (V(I,I) .GT. 0.D0) GO TO 228
      WRITE (ISYSWR,510) I
      MDIAG = 1
228  R = 0.D0
      DO 240 J= 1, NPAR
      IF (I .EQ. J) GO TO 230
      IF (V(I,J)**2 .LT. DABS(V(I,I)*V(J,J))) GO TO 230
      WRITE (ISYSWR, 470) I,J
      V(I,J) = 0.D0
      V(J,I) = 0.D0
230  CONTINUE
240  R = R + V(I,J) * G(J)
250  SIGMA = SIGMA + 0.500 *R *G(I)
      IF (MDIAG .EQ. 1) GO TO 390
      IF (SIGMA .GT. 0.D0) GO TO 400

```

```

WRITE (ISYSWR,530)
GO TO 216
390 ISW(2) = 0
400 RETURN
460 FORMAT(10H0PARAMETER I3,2H, A10, 16H, HAS BEEN FIXED 9X 19HFIRS
1T DERIVATIVE IS E11.3/1X40(1H*),8X 20HSECOND DERIVATIVE IS E11.3
2/)
470 FORMAT ( 71H COVARIANCE MATRIX NOT POSITIVE-DEFINITE. FAULTY E
1LEMENT IN POSITION2I3)
480 FORMAT ( 35H SECOND DERIVATIVE MATRIX INVERTED )
490 FORMAT ( 76H0CALL LIMIT IN HESSE. OFF-DIAGONAL ELEMENTS CALCULAT
1ED ONLY UP TO POSITION2I3/)
500 FORMAT ( 37H START SECOND DERIVATIVE CALCULATION )
510 FORMAT ( 17H DIAGONAL ELEMENT I5, 20H IS ZERO OR NEGATIVE)
520 FORMAT ( 23H MATRIX INVERSION FAILS)
530 FORMAT ( 29H MATRIX NOT POSITIVE-DEFINITE)
540 FORMAt ( 30H ONLY DIAGONAL MATRIX PRODUCED)
END
C*****IBS*****
SUBROUTINE IBS(M)
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'BINFF.CCC'
INCLUDE 'BMI1.CCC'
INCLUDE 'FILES.CCC'
C
COMMON/IBSDAT/ A(40),BTX(512),BTY(512),APX(512),APY(512),APP(512),
1 AP(512),AA(512),BB(512),CC(512),BPAR(10),SS(512)
C
C
C .....
C PWSK IBS KA KB BML
C A(1) A(2) .... A(6)
C .....
C
C KA = 11,12 PRINT/NO PRINT OF TABLE
C (KA=0 DEFAULTS TO KA=11— PRINT)
C KB = NUMBER OF SUPERPERIODS. NEGATIVE FOR REFLECTION, AS IN CYC.
C BML BEAM LINE
C A(1) = GAMMA A(4) = SQRT(2) SIGMA BETA GAMMA
C A(2) = EPSILON X A(5) = GAMMA S
C A(3) = EPSILON Y A(6) = NO. OF PARTICLES IN BUNCH
C
C FILE NAMED JBIS IN COMMON BLOCK FILES IS SET IN BLOCK DATA.
C FILE JBIS IS MADE BY CYX.
CALL MIFILL (M,1,1,NELS1,M11)
KA = INFF(4,M)
IF (KA.EQ.0) KA=11
INFF(4,M) = KA
CALL CYX(M)
C
IJOE = NELS1 +1
C*****
C INPUT LATTICE PARAMETERS
C*****
DO 99 JOE=1,IJOE

```



```

      READ(JBIS) X,X,X,SS(JOE),X,X,X,X,X,X,BTY(JOE),BTX(JOE),APY(JOE),
1      APX(JOE),X,X,AP(JOE),APP(JOE)
99  CONTINUE
C*****
C  RETRIEVE BEAM GAMMA, NORMALIZED X AND Y EMITTANCE, SIGMABETAGAMMA,
C  SIGMA LENGTH AND NUMBER OF PARTICLES.
C*****
      CALL DATA(M,1,1,6,A(2))
      CALL PWNSKY(IJOE)
      RETURN
      END

C ***** IDAT *****
      FUNCTION IDAT(M,J)
      IMPLICIT REAL*8 (A-H,O-Z)
C  RETRIEVES ONE PIECE OF INTEGER DATA FROM INDEX J OF ELEMENT M
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      LOCI=INFF(12,M)*2-1
      LOC = LOCI+J-1
      IDAT = ISTORE( LOC)
      RETURN
      END

C ***** IMPROV *****
      SUBROUTINE IMPROV
      IMPLICIT REAL*8 (A-H,O-Z)
CC      ATTEMPTS TO IMPROVE ON A GOOD LOCAL MINIMUM BY FINDING A
CC      BETTER ONE.  THE QUADRATIC PART OF FCN IS REMOVED BY CALFCN
CC      AND THIS TRANSFORMED FUNCTION IS MINIMIZED USING THE SIMPLEX
CC      METHOD FROM SEVERAL RANDOM STARTING POINTS.
CC      REF. — GOLDSTEIN AND PRICE, MATH.COMP. 25, 569 (1971)
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAUT ,SIGMA
      COMMON/CASC/ JH, JL, Y(16)
      COMMON /DERIVA/ G(30) ,G2(30)
      COMMON /SIMVEC/ P(15,16),PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15)
      COMMON /VARIAT/ VT(15,15)
      INTEGER CWORD,CWORD2,CWORD3
      REAL*4 RANG
      DATA ALPHA,BETA,GAMMA / 1.0D0, 0.5D0 , 2.0D0 /
      IF (NPAR .LE. 0) RETURN
      ITAUT = 1
      EPSI = 0.1D0*UP
      NPFN=NFCN
      NLOOP = WORD7(2)
      IF (NLOOP .LE. 0) NLOOP = NPAR + 4

```

```

NPARP1=NPAR+1
WG = 1.0D0/DFLOAT(NPAR)
IFLAG=4
SIGSAV = SIGMA
APSI = AMIN
DO 2 I= 1, NPAR
  XT(I) = X(I)
  G2(I) = DSQRT(V(I,I)*UP)
DO 2 J= 1, NPAR
  2 VT(I,J) = V(I,J)
  CALL VERMIN(VT,MAXINT,MAXINT,NPAR,IFAIL)
  IF (IFAIL .GE. 1) GO TO 280
  LOOP = 0
  3 DO 4 I= 1, NPAR
    DIRIN(I) = 2.0D0*G2(I)
  4 X(I)= XT(I)+2.0D0*DIRIN(I)*(DBLE(RAN6(0))-0.5D0)
    LOOP = LOOP + 1
    REG = 2.0D0
    WRITE (ISYSWR, 1040) LOOP
  5 AMIN = CALFCN(X)

```

C SET UP RANDOM SIMPLEX

```

  JL = NPARP1
  JH = NPARP1
  Y(NPARP1) = AMIN
  AMAX = AMIN
  DO 15 I= 1, NPAR
    XI = X(I)
    X(I)= XI-DIRIN(I)*(DBLE(RAN6(0))-0.5D0)
    Y(I) = CALFCN(X)
    IF (Y(I) .GE. AMIN) GO TO 7
    AMIN = Y(I)
    JL = I
  7 IF (Y(I) .LE. AMAX) GO TO 8
    AMAX = Y(I)
    JH = I
  8 CONTINUE
  DO 10 J= 1, NPAR
  10 P(J,I) = X(J)
    P(I,NPARP1) = XI
  15 X(I) = XI
    SIGMA = AMIN
    SIG2 = SIGMA

```

C START MAIN LOOP

```

  50 CONTINUE
    IF (AMIN .LT. 0.D0) GO TO 95
    IF (ISW(2) .LT. 2) GO TO 280
    EP = 0.1D0*AMIN
    IF (SIG2 .LT. EP .AND. SIGMA.LT.EP ) GO TO 100
    SIG2 = SIGMA
    IF ((NFCN-NPFN) .GT. NFCNMX) GO TO 300

```

C CALCULATE NEW POINT * BY REFLECTION

```

  DO 60 I= 1, NPAR
    PB = 0.D0
  DO 59 J= 1, NPARP1
  59 PB = PB + WG * P(I,J)

```

```

    PBAR(I) = PB - WG * P(I,JH)
60 PSTAR(I)=(1.0D0+ALPHA)*PBAR(I)-ALPHA*P(I,JH)
    YSTAR = CALFCN(PSTAR)
    IF(YSTAR.GE.AMIN) GO TO 70
C    POINT * BETTER THAN JL, CALCULATE NEW POINT **
    DO 61 I=1,NPAR
61 PSTST(I)=GAMMA*PSTAR(I)+(1.0D0-GAMMA)*PBAR(I)
    YSTST = CALFCN(PSTST)
66 IF (YSTST .LT. Y(JL)) GO TO 67
    CALL RAZZIA(YSTAR,PSTAR)
    GO TO 50
67 CALL RAZZIA(YSTST,PSTST)
    GO TO 50
C    POINT * IS NOT AS GOOD AS JL
70 IF (YSTAR .GE. Y(JH)) GO TO 73
    JHOLD = JH
    CALL RAZZIA(YSTAR,PSTAR)
    IF (JHOLD .NE. JH) GO TO 50
C    CALCULATE NEW POINT **
73 DO 74 I=1,NPAR
74 PSTST(I)=BETA*P(I,JH)+(1.0D0-BETA)*PBAR(I)
    YSTST = CALFCN(PSTST)
    IF(YSTST.GT.Y(JH)) GO TO 5
C    POINT ** IS BETTER THAN JH
    IF (YSTST .LT. AMIN) GO TO 67
    CALL RAZZIA(YSTST,PSTST)
    GO TO 50
C    . . . . . END MAIN LOOP
95 WRITE (ISYSWR,1000)
    REG = 0.1D0
C    . . . . . ASK IF POINT IS NEW
100 CALL INTOEX(X)
    CALL FCN(NPAR,G,AMIN,U,4)
    NFCN = NFCN + 1
    DO 120 I= 1, NPAR
    DIRIN(I) = REG*G2(I)
    IF (DABS(X(I)-XT(I)) .GT. DIRIN(I)) GO TO 150
120 CONTINUE
    GO TO 230
150 NFCNMX = NFCNMX + NPFN - NFCN
    NPFN = NFCN
    CALL SIMPLX
    IF (AMIN .GE. APSI) GO TO 325
    DO 220 I= 1, NPAR
    DIRIN(I) = 0.1D0 *G2(I)
    IF (DABS(X(I)-XT(I)) .GT. DIRIN(I)) GO TO 250
220 CONTINUE
230 IF (AMIN .LT. APSI) GO TO 350
    GO TO 325
C    . . . . . TRULY NEW MINIMUM
250 NEWMIN = 1
    ISW(2) = 0
    ITAUR = 0
    NFCNMX = NFCNMX + NPFN - NFCN
    WRITE (ISYSWR,1030)

```

```

      RETURN
C      . . . RETURN TO PREVIOUS REGION
280 WRITE (ISYSWR,1020)
     ISW(2) = 0
     GO TO 325
300 ISW(1) = 1
325 DO 330 I= 1, NPAR
     DIRIN(I) = 0.01D0*G2(I)
330 X(I) = XT(I)
     AMIN = APSI
     SIGMA = SIGSAV
350 CALL INTOEX(X)
     WRITE (ISYSWR,1010)
     IF (ISW(2) .LT. 2) GO TO 380
     IF (LOOP .LT. NLOOP .AND. ISW(1) .LT. 1) GO TO 3
     ISW(2) = 3
380 CALL MPRINT (1,AMIN)
     RETURN
1000 FORMAT ( 54H AN IMPROVEMENT ON THE PREVIOUS MINIMUM HAS BEEN FOUN
1D)
1010 FORMAT ( 51H IMPROVE HAS RETURNED TO REGION OF ORIGINAL MINIMUM)
1020 FORMAT ( 44H0COVARIANCE MATRIX WAS NOT POSITIVE-DEFINITE)
1030 FORMAT ( 38H0IMPROVE HAS FOUND A TRULY NEW MINIMUM/1H 37(1H*))
1040 FORMAT ( 18H0START ATTEMPT NO.12, 20H TO FIND NEW MINIMUM)
     END
C ***** INCR *****
      SUBROUTINE INCR(M,KA)
      IMPLICIT REAL*8 (A-H,O-Z)
C CARD IMAGE      INCR      K      // Q      TYPE      DELTA
C INCREMENT THE KTH PARAMETER OF Q BY DELTA
C TYPE MAY BE F, I, KA OR KB.
C DELTA WILL BE FL. PT. IF TYPE IS F OR OMITTED.
C IT WILL BE INTEGER IF I, KA OR KB.
C IF MODE=1,THE NEW VALUE REMAINS IN DATA STORAGE
C IF MODE=3,THE ORIGINAL VALUE IS REPLACED AFTER REQUIRED ITERATIONS
C HAVE BEEN COMPLETED.
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))

      DATA IBLNK/'      '/
      INTEGER doIF/'F '/,doII/'I '/,doIKA/'KA'/,doIKB/'KB'/
      INTEGER BDAT
      INTEGER ELNUM

C
C FIND NAME AND INDEX OF ELEMENT WHOSE DATA IS TO BE INCREMENTED.
C DETERMINE TYPE OF DATA TO BE CHANGED — FL. PT., INTEGER, KA OR KB.
      NM = BDAT(M,1)
      N = ELNUM(NM)
      ITP = BDAT(M,2)
      IF ((ITP.EQ.doIF).OR.(ITP.EQ.IBLNK))IGO = 1
      IF (ITP.EQ.doII) IGO = 2

```

```

      IF (ITP.EQ.doIKA) IGO = 3
      IF (ITP.EQ.doIKB) IGO = 4
C      IPP NOT EQUAL TO 0, SUPPRESSES PRINT
      IPP = INFF(6,N)
      IF (MODE.EQ.1) GO TO 30
C      MS IS M OF THE SUBROUTINE THAT INCR IS IN.
      MS = INFF(19,M)
C      DETERMINE IF THIS IS FIRST AND/OR LAST ITERATION THROUGH SUBROUTINE.
      IF (IDAT(MS,1).EQ.1) GO TO 50
20     IF (IDAT(MS,2).EQ.1) GO TO 60
C      INCREMENT PARAMETER, RETURN NEW VALUE TO DATA STORE
30     CONTINUE
      GO TO (31,32,33,34), IGO
C      FLOATING POINT
31     DF = FDAT(M,1)
      CALL AFD(N,KA,DF)
      F = FDAT(N,KA)
      GO TO 40
C      INTEGER
32     IDF = IDAT(M,1)
      INDAT=IDAT(N,KA)
      INEW = IDF + INDAT
      CALL REPINT(N,KA,INEW)
      GO TO 40
C      INCREMENT KA OR KB
33     JI = 4
      GO TO 35
34     JI = 5
35     IDF = IDAT(M,1)
      INDAT = INFF(JI,N)
      INEW = IDF + INDAT
      INFF(JI,N) = INEW
40     KO = KA
      IF (IPP.GT.0) RETURN
      GO TO (41,42,43,44), IGO
41     WRITE (3,1) KO,NM,DF,F
      RETURN
42     WRITE (3,2) KO,NM,IDF,INEW
      RETURN
43     WRITE (3,3) NM,IDF,INEW
      RETURN
44     WRITE (3,4) NM,IDF,INEW
      RETURN
C      SAVE ORIGINAL PARAMETER
50     CONTINUE
      GO TO (51,52,53,54), IGO
51     SF = FDAT(N,KA)
      CALL REPFLT(M,2,SF)
      GO TO 20
52     ISF = IDAT(N,KA)
      GO TO 55
53     ISF = INFF(4,N)
      GO TO 55
54     ISF = INFF(5,N)
55     CALL REPINT(M,2,ISF)

```

```

      GO TO 20
C      ITERATIONS COMPLETED.  RESTORE ORIGINAL PARAMETER.
60    CONTINUE
      GO TO (61,62,62,62), IGO
61    SF = FDAT(M,2)
      CALL REPFLT(N,KA,SF)
      F = SF
      DF = 0.
      GO TO 40
62    ISF = IDAT(M,2)
      GO TO (63,63,64,65), IGO
C  SHOULD NOT GET HERE
63    WRITE (3,5) IGO
      ERROR = .TRUE.
      RETURN
64    JI = 4
      GO TO 66
65    JI = 5
66    INFF(JI,N) = ISF
      INEW = ISF
      IDF = 0
      GO TO 40
C *****FORMATS*****
1    FORMAT (6H  *** ,8X,5HINCR ,1X,I3,4X,5H // ,A5,5X,F10.6,5X,
1  9H VALUE = ,F14.6)
2    FORMAT (6H  *** ,8X,5HINCR ,1X,I3,4X,5H // ,A5,2X,I10,5X,
1  9H VALUE = ,I10)
3    FORMAT (6H  *** ,8X,5HINCR ,2X,2HKA,4X,5H // ,A5,5X,I10,5X,
1  9H VALUE = ,I10)
4    FORMAT (6H  *** ,8X,5HINCR ,2X,2HKB,4X,5H // ,A5,5X,I10,5X,
1  9H VALUE = ,I10)
5    FORMAT (6H  *** , 'ERROR IN INCR ROUTINE.', I10, 'IS INVALID COMPUTE
1ED GO TO.')
      END
C ***** INFW *****
      SUBROUTINE INFW(IA,J,M)
      IMPLICIT REAL*8 (A-H,O-Z)
C  WRITES ONE INTEGER WORD FROM I INTO INFF(J,M)
      INCLUDE 'BINFF.CCC'
      INFF(J,M) = IA
      RETURN
      END
C ***** INP *****
      SUBROUTINE INP(M,KIND,NQ,N1,I1X,N2X,I2X,N3X,I3X,N4X,I4X,N5X,I5X,
      *   N6X,I6X,N7X,I7X,N8X,I8X,N9X,I9X,N10X,I10X)
      IMPLICIT REAL*8 (A-H,O-Z)
C  RESERVES AND STORES INPUT
      INCLUDE 'CONTRL.CCC'

      I1 = I1X
      N2 = N2X
      I2 = I2X
      N3 = N3X
      I3 = I3X
      N4 = N4X

```

```

I4 = I4X
N5 = N5X
I5 = I5X
N6 = N6X
I6 = I6X
N7 = N7X
I7 = I7X
N8 = N8X
I8 = I8X
N9 = N9X
I9 = I9X
N10 = N10X
I10 = I10X
GO TO 2

```

C

```

ENTRY INPA(M,KIND,NQ,N1,I1X,N2X,I2X,N3X,I3X,N4X)
I1 = I1X
N2 = N2X
I2 = I2X
N3 = N3X
I3 = I3X
N4 = N4X
IF (N4 .EQ. 0) GO TO 2
WRITE(3, 1004) N4, M
1004 FORMAT(' INPA ERROR', 2I6)
GO TO 1

```

C

```

ENTRY INP8(M,KIND,NQ,N1,I1X,N2X,I2X,N3X)
I1 = I1X
N2 = N2X
I2 = I2X
N3 = N3X
IF (N3 .EQ. 0) GO TO 2
WRITE(3, 1005) N3, M
1005 FORMAT(' INP8 ERROR', 2I6)
GO TO 1

```

C

```

ENTRY INP6(M,KIND,NQ,N1,I1X,N2X)
I1 = I1X
N2 = N2X
IF (N2 .EQ. 0) GO TO 2
WRITE(3, 1006) N2, M
1006 FORMAT(' INP6 ERROR', 2I6)
GO TO 1

```

C

```

ENTRY INP4(M,KIND,NQ,N1)
IF (N1 .EQ. 0) GO TO 2
WRITE(3, 1007) N1, M
1007 FORMAT(' INP4 ERROR', 2I6)

```

C

```

1 CONTINUE
WRITE(3, 1008) M, KIND, NQ
WRITE(3, 1008) N1,N2,N3,N4,N5,N6,N7,N8,N9,N10
WRITE(3, 1008) I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
1008 FORMAT(' ', 10I6)

```

```

c    CALL ERRTRA
    CALL EXIT
C
2 CONTINUE
C
    IF (RSRV) CALL RESRV(M,KIND,NQ,N1,I1,N2,I2,N3,I3,N4,I4,N5,I5,N6,I6
1,N7,I7,N8,I8,N9,I9,N10,I10)
    IF(ERROR) GO TO 10
    IF (STOR) CALL LOAD(M,N1,I1,N2,I2,N3,I3,N4,I4,N5,I5,N6,I6,N7,I7,N8
1,I8,N9,I9,N10,I10)
10 CONTINUE
    RETURN
    END
C ***** INTCON *****
    SUBROUTINE INTCON(N,NA)
    IMPLICIT REAL*8 (A-H,O-Z)

C    NA IS AN ARRAY OF LENGTH N.  IT CONTAINS NO MORE THAN 4 BCD
C    CHARACTERS, REPRESENTING AN INTEGER WITHIN PARENTHESES.
C    ROUTINE EXTRACTS THE INTEGER AND RETURNS IT IN THE ARRAY IN
C    INTEGER FORM.

    DIMENSION NA(1)
    INTEGER doIBLANK/'  '/'

    DO 100 I=1,N
    DECODE(4,1,NA(I))NB
1    FORMAT(3X,A1)
    IF (NB.EQ.doIBLANK) GO TO 50
C        ASSUMES 1 DIGIT INTEGER
    READ (99,2) NB
2    FORMAT (1X,I1,2X)
    GO TO 70
C        ASSUMES 2 DIGIT INTEGER
50    CONTINUE
    READ (99,3)NB
3    FORMAT (1X,I2,1X)
70    NA(I) = NB
100 CONTINUE
    RETURN
    END
C ***** INTOEX *****
    SUBROUTINE INTOEX(PINT)
    IMPLICIT REAL*8 (A-H,O-Z)
CC    TRANSFORMS FROM INTERNAL COORDINATES (PINT) TO EXTERNAL
CC    PARAMETERS (U).  THE MINIMIZING ROUTINES WHICH WORK IN
CC    INTERNAL COORDINATES CALL THIS ROUTINE BEFORE CALLING FCN.
    INCLUDE 'PAREXT.CCC'
    COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
    DIMENSION PINT(2)
    DO 100 I= 1, NU
    J = LCORSP(I)
    IF ( J ) 100,100,50
50 CONTINUE
    IF (LCODE(I) .EQ. 1) GO TO 80

```



```

      AL = ALIM(I)
      U(I) = AL + 0.5D0 *(DSIN(PINT(J)) + 1.0D0) * (BLIM(I) -AL)
      GO TO 100
    80 U(I) = PINT(J)
  100 CONTINUE
      RETURN
      END

C ***** INV *****
      SUBROUTINE INV(M,N)
      IMPLICIT REAL*8 (A-H,O-Z)
C RETRIEVES RX AND RY MATRICES FROM LQ(N), INVERTS THEM AND STORES THEM
C IN LQ(M). DIMENSIONS ARE (2,3) FOR EACH
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BWORK.CCC'
      INCLUDE 'SWTCH.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      DIMENSION T(49),RW(3),RX(2,3),RY(2,3),RPQ(4)
      INTEGER doIR7/'R7' '/',doIROT/'ROT' '/'
      K1 = INFF(20,M)
C REVERSE SIGN FOR INVERSE RETRIEVE
      NN = - N
C      INVERT MATRIX AS 7X7, THEN STORE ACCORDING TO KIND
  10 CALL RTRV7(NN,T,RW)
      IF(K1.EQ.doIR7.OR.K1.EQ.doIROT) GO TO 15
      KN = 3
      CALL CHVMAT(KN,T,RX,RY,RW,RPQ)
      CALL STXY(M,RX,RY,RW)
      CALL STDAT(M,5,4,4,RPQ)
      RETURN
  15 CALL STOR7(M,T,RW)
      RETURN
      END

C ***** INV2 *****
      SUBROUTINE INV2(M,MB)
      IMPLICIT REAL*8 (A-H,O-Z)

C ROTATE MATRIX MB BY 180 DEGREES, REFLECT IT AND STORE IN M.
C M AND MB ARE INDECESES OF MATRICES

      INCLUDE 'DIM.CCC'
      INCLUDE 'CONST.CCC'
      DIMENSION RX(2,3),RY(2,3),RW(3),T(49),TT(7,7)
      EQUIVALENCE (T(1),TT(1))

      CALL ROTM(M7END,MB,PI)
      NN = M7END
      CALL REF(M,NN)
      RETURN
      END

C ***** KICK *****
      SUBROUTINE KICK(M)
      IMPLICIT REAL*8 (A-H,O-Z)
C H KICK K // C
      BRHO BK

```

```

C BK = KICKER FIELD
C K = 1,2 FOR FIELD IN 1,2 (X,Y) DIRECTION
C C = PREVIOUSLY DEFINED DRF OR MAG
C IF C IS A DRF OF ZERO LENGTH, BK = KICKER LENGTH X FIELD
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BWORK.CCC'
  INCLUDE 'BINFF.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))
  DIMENSION T(7,7),RW(3)
  INTEGER OPNAME
  INTEGER doIDRF/'DRF '/
  common/seed/nr
  N=MDAT(M,1)
  CALL RTRV7(N,T,RW)
  BK=FDAT(M,3)
  BKK=BK
C IF (KB.GE.1) ERROR IS RANDOM WITH GIVEN RANGE
  KB=INFF(5,M)
  IF (KB.GE.1) BK=BKK*(ran(nr)-.5)
  IF (BK.EQ.0.) GO TO 4
  K=INFF(4,M)
  OPNAME=INFF(1,N)
  IF (OPNAME.NE.doIDRF) GO TO 3
2  S=FDAT(N,1)
  BRHO=FDAT(M,2)
  IF (S.NE.0.) GO TO 5
  RHO=0.
  THETA=BK/BRHO
  XK=0.
  GO TO 6
5  CONTINUE
  RHO=BRHO/BK
  THETA=S/RHO
  XK = -RHO*THETA*THETA/2.
6  DXK = -THETA
  T(5,7)=XK*THETA/3.
  T(5,6)=-2.*T(5,7)
  RW(3)=T(5,6)
  IF (K.NE.1) GO TO 1
  T(3,6)=XK
  T(3,7)=-XK
  T(4,6)=DXK
  T(4,7)=-DXK
  T(5,3)=-DXK
  T(5,4)=-XK
  GO TO 4
1  T(1,7)=XK
  T(1,6)=-XK
  T(2,6)=-DXK
  T(2,7)=DXK
  T(5,1)=DXK
  T(5,2)=XK
  RW(2)=THETA
  GO TO 4

```

```

3   B0=FDAT(N,4)
    IF ( (B0.EQ.0.) .OR. (K.EQ.1) ) GO TO 2
    DBB=BK/B0
    T(1,7)=-T(1,6)*DBB
    T(2,7)=-T(2,6)*DBB
4   CALL STOR7(M,T,RW)
    BK=BKK
    RETURN
    END

C ***** KINEM *****
    SUBROUTINE KINEM(KP,T,P)
    IMPLICIT REAL*8 (A-H,O-Z)
C   KP IS TYPE OF PARTICLE, T IS KE, P IS BETGAM
    DIMENSION ER(2)
    DATA ER /.511006D0,938.256D0/
    E = ER(KP) + T
    GAM = E/ER(KP)
    P = DSQRT(GAM**2-1.)
    BET = P/GAM
    RETURN
    END

C ***** LOAD *****
    SUBROUTINE LOAD(M0,N1,I1,N2,I2X,N3X,I3X,N4X,I4X,N5X,I5X,
1      N6X,I6X,N7X,I7X,N8X,I8X,N9X,I9X,N10X,I10X)

    IMPLICIT REAL*8 (A-H,O-Z)

C
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'BWORK.CCC'
    DIMENSION ISTORE(1)
    EQUIVALENCE (STORE(1),ISTORE(1))
    INCLUDE 'INSTR.CCC'
    INCLUDE 'STORE.CCC'
    INCLUDE 'CONTRL.CCC'
    INCLUDE 'FLTN.CCC'
    INCLUDE 'DIM.CCC'
    INCLUDE 'COPY.CCC'
    INCLUDE 'FILES.CCC'
    DIMENSION IFCD(12)
    INTEGER BDAT
    DIMENSION IF(15),IB(15),FCD( 6 ),BCD( 12),ICD(12 ),ND(10),IT(10),
1 II(15)
    DIMENSION IF1(6), IF2(6), IF3(3)
    INTEGER*2 IFd0I2(30), IFd0I3(6)
    EQUIVALENCE (IF(1), IFd0I2(1))
    EQUIVALENCE (IF3(1), IFd0I3(1))
    EQUIVALENCE( IF(1),ICARD(6) )
    INTEGER BCD,OPN,RET
    INTEGER SHFTR, SHFTL
    EQUIVALENCE (FCD(1),IFCD(1))
    INTEGER HCOPY
    DATA HCOPY/4HCOPY/
    DATA MASKF/Z000000FF/
    DATA IBLANK/'      '/

```

```

INTEGER DOLEVEC/'EVEC'//,DOLEQIL/'EQIL'//,DOLEQUAL/'=' '//,
1      doISCAL/'SCAL'//,doIR7/'R7' '//,doIROT/'ROT' '//,
2      doIMXY/'MXY' '//,doISNGL/'SNGL'//,doICYB/'CYB' '//,
3      doINOSV/'NOSV'//,doIREM/'REM'//,doIPAGE/'PAGE'//

```

C

```

I2 = I2X
N3 = N3X
I3 = I3X
N4 = N4X
I4 = I4X
N5 = N5X
I5 = I5X
N6 = N6X
I6 = I6X
N7 = N7X
I7 = I7X
N8 = N8X
I8 = I8X
N9 = N9X
I9 = I9X
N10 = N10X
I10 = I10X

```

```
GO TO 601
```

```
ENTRY LOADA(M0,N1,I1,N2,I2X,N3X,I3X,N4X,I4X,N5X)
```

```
I2 = I2X
```

```
N3 = N3X
```

```
I3 = I3X
```

```
N4 = N4X
```

```
I4 = I4X
```

```
N5 = N5X
```

```
IF (N5 .EQ. 0) GO TO 601
```

```
WRITE (3, 1014) N5, M0
```

```
1014 FORMAT(' LOADA ERROR', 2I6)
```

```
GO TO 600
```

```
ENTRY LOAD8(M0,N1,I1,N2,I2X,N3X,I3X,N4X)
```

```
I2 = I2X
```

```
N3 = N3X
```

```
I3 = I3X
```

```
N4 = N4X
```

```
IF (N4 .EQ. 0) GO TO 601
```

```
WRITE (3, 1015) N4, M0
```

```
1015 FORMAT(' LOAD8 ERROR', 2I6)
```

```
GO TO 600
```

```
ENTRY LOAD6(M0,N1,I1,N2,I2X,N3X)
```

```
I2 = I2X
```

```
N3 = N3X
```

```
IF (N3 .EQ. 0) GO TO 601
```

```
WRITE (3, 1016) N3, M0
```

```
1016 FORMAT(' LOAD6 ERROR', 2I6)
```

```
GO TO 600
```

```
ENTRY LOAD4(M0,N1,I1,N2)
```

```
IF (N2 .EQ. 0) GO TO 601
```

```
WRITE (3, 1017) N2, M0
```

```
1017 FORMAT(' LOAD4 ERROR', 2I6)
```

```

600 CONTINUE
  WRITE (3, 1018) N1,N2,N3,N4,N5,N6,N7,N8,N9,N10
  WRITE (3, 1018) I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
1018 FORMAT(' ', 10I6)
  CALL EXIT
601 CONTINUE
  M = M0
  IF( .NOT.STOR ) RETURN
  IF( MODE.EQ.3 ) RETURN
  CALL ARGINP( N1,I1,N2,I2,N3,I3,N4,I4,N5,I5,N6,I6,N7,I7,
1          N8,I8,N9,I9,N10,I10,ND,IT,NSETS )
  RET=1
  IF (.NOT.EMPTY) GO TO 1
  NI = 0
  IF(.NOT.INDEF) GO TO 500
  ND(NSETS)=24
  IT(NSETS)=-IT(NSETS)
  NS = 0
500  IPRNT=0
  NB = 0
  NF = 0
  EMPTY = .FALSE.
  LDFLG=.TRUE.
  GO TO 3
C
1  READ(2,100) ( ICARD(IW),IW=1,20)
100  FORMAT(20A4)
3  CONTINUE
  IF (ICARD(1).EQ.IBLANK.AND.ICARD(2).EQ.IBLANK) GO TO 201
  DECODE(20,101,ICARD)ISGN,NM,OPN,KAM,KBM
101  FORMAT (A1,A4,2X,A4,1X,A3,1X,A3,1X)
  IF (.NOT.CPYSW) GO TO 501
  IF(IPRNT.NE.0) GO TO 501
  WRITE (3,106) ISGN,NM,OPN,KAM,KBM,(ICARD(IW),IW=6,20)
106  FORMAT (6H  *** ,A1,A5,2X,A5,1X,A3,1X,A3,5H // ,15A4)
  IPRNT=IPRNT+1
501  CONTINUE
  IF (.NOT.(NS.EQ.NSETS.AND.INDEF)) GO TO 202
  GO TO 74
201  IF (.NOT.CPYSW) GO TO 202
  WRITE (3,107) (ICARD(IW),IW=6,20)
107  FORMAT (6H  * ,21X,5H // ,15A4)
202  IF (NSETS.EQ.0) RETURN
  DECODE(60,102,IF)(IB(J),J=1,12)
102  FORMAT (12(A4,1X))
  DECODE(60,105,IF)(II(J),J=1,12)
105  FORMAT (12(1X,A4))
  DECODE(60,109,IF)(IF1(J),IF2(J),J=1,6)
109  FORMAT(6(A1, A4, 5X))
  NI0 = 0
  NB0 = 0
  NF0 = 0
  I = 1
  J = 1
  GO TO (2,5,6,7,8,9), RET

```

```

C.....
2  CONTINUE
   DO 4 NS=1,NSETS
   RET=2
   IF ( J.GT.12) GO TO 1
5  CONTINUE
   K = 1
   ITNS=IT(NS)
   GO TO ( 6,7,8,9 ), ITNS
C
C FLOATING POINT
C TEST FOR SYMBOLIC FLOATING POINT
6  CONTINUE
   MSK = LAND(MASKF,IF2(I))
   MSK=SHFTL(MSK,24)
   DO 10 L=1,14
   IF (MSK.EQ.IFL(L)) GO TO 120
10  CONTINUE
12  continue
   ifcd(2*nf0+1)=0
   ifcd(2*nf0+2)=ib(2*i-1)
   go to 110
120 CONTINUE
   MSK=LAND(MASKF,IF1(I))
   msk=shftl(msk,24)
   DO 121 L = 1,14
       IF (MSK .EQ. IFL(L)) GO TO 122
121 CONTINUE
   GO TO 12
C FLOATING POINT NUMBER
122 CONTINUE
   L = 5*I -4
   DO 13 N = 1,5
       IFdo13(N) = IFdo12(L)
       L = L +1
13  CONTINUE
   DECODE(12,103,IF3)FCD(NF0+1)
103 FORMAT(F10.5, 2X)
110 K = K + 1
   I=I+1
   J=J+2
   NF0=NF0+1
   IF( K.GT.ND(NS) ) GO TO 61
   IF( I.LE.6 ) GO TO 6
   CALL STDAT( M,1,NF + 1,NF0,FCD )
   NF = NF + NF0
   RET=3
   GO TO 1
61  CALL STDAT(M,1,NF+1,NF0,FCD)
   NF=NF+NF0
   NF0=0
   GO TO 4
C
C BCD
7  CONTINUE

```

```

BCD(NB0+1) = IB(J)
K = K + 1
J = J + 1
I = J/2 + 1
NB0 = NB0 + 1
IF(K.GT.ND(NS)) GO TO 73
IF( J.LE.12) GO TO 7
CALL STDAT( M,2,NB + 1,NB0,BCD )
NB = NB + NB0
RET=4
IF (NS.NE.NSETS) GO TO 1
IF (INDEF) ND(NS)=ND(NS)+12
GO TO 1
73  J = 2*(J/2) + 1
    CALL STDAT(M,2,NB+1,NB0,BCD)
    NB=NB+NB0
    NB0=0
    GO TO 4

```

C LAST SET WAS BCD OF INDEFINITE LENGTH
 C WARNING—AS PRESENTLY CODED(7/1975) , NO INTEGER SETS MAY BE
 C INCLUDED. TO BE SAFE, THERE SHOULD ONLY BE A SINGLE SET LOADED.
 C E.G. CALL INP(M,MXPMY,-0,1,-2,0)
 74 CONTINUE

```

EMPTY=.TRUE.
INDEF=.FALSE.
NBC=NB
LFMX=LBC+NBC
NTOT=LFMX-LFILE
INFF(17,M)=NBC
INFF(18,M)=NTOT
LFILE=LFMX
IF(LFILE.LE.LMAX) GO TO 340
WRITE(3,75)
NSTO = LMAX - LFILE
NINFF = MAX - MIN
WRITE (3,76) LMAX,LFILE,NSTO,MAX,NINFF,M
75  FORMAT (6H0*****, 'STORE OVERFLOWED.'
1  ' INCREASE STORE ARRAY AND LMAX.')
76  FORMAT(/1X'CORE USE SUMMARY',27X,'MAXIMUM',17X,'USED',8X,'UNUSED'/
1 34X,'STORE (ELEMENT STORAGE)',14X,I10,1X,'(LMAX)',4X,I10,5X,I10/
2 34X,'INFF (ELEMENT DEFINITIONS)',10X,I10,1X,'(MAX)',5X,I10,5X,
3 I10)
ERROR=.TRUE.
GO TO 340

```

C
 C INTEGER
 8 CONTINUE
 DECODE(4,104,II(J))ICD(NI0+1)
 104 FORMAT (I4)
 K = K + 1
 82 J = J + 1
 I = J/2 + 1

```

      NI0 = NI0 + 1
      IF(K.GT.ND(NS)) GO TO 83
      IF ( J .LE.12) GO TO 8
      CALL STDAT( M,3,NI + 1,NI0,ICD )
      NI = NI + NI0
      RET=5
      GO TO 1
83    J = 2*(J/2) + 1
      CALL STDAT(M,3,NI+1,NI0,ICD)
      NI=NI+NI0
      NI0=0
      GO TO 4
C
C  SKIP ND(NS) WORDS
9    CONTINUE
      K = K + 1
      I = I + 1
      J = J + 2
      IF( K.GT.ND(NS)) GO TO 4
      IF ( I.LE.6 ) GO TO 9
      RET=6
      GO TO 1
4    CONTINUE
C.....
      IF (OPNAME.EQ.doIVEEC.OR.OPNAME.EQ.doIEQIL) GO TO 310
      IF (KIND.NE.IBLANK) GO TO 340
C    KIND HAS TO BE DETERMINED BY BCD INPUT.
      IJ = 1
      NCD=IABS(MDAT(M,1))
      IF (NCD.NE.0) KND=INFF(20,NCD)
      IF (OPNAME.EQ.doIEQUAL) IJ = 2
      IF (OPNAME.EQ.doIEQUAL.AND.KND.EQ.doISCAL) GO TO 315
      DO 305 J=1,NBC,IJ
      NCD=IABS(MDAT(M,J))
C  IF ELEMENT NO.=0,IT HAS NOT YET BEEN DEFINED
      IF (NCD.EQ.0) GO TO 320
307   KND = INFF(20,NCD)
      IF (KND.EQ.doIR7.OR.KND.EQ.doIROT) GO TO 320
305   CONTINUE
C  MAKE ELEMENT A MXY KIND
      INFF(20,M) = doIMXY
      ITE = 1
      ITYP = INFF(21,NCD)
      INFF(21,M)=ITYP
      IF(ITYP.NE.doISNGL) ITE=2
      IF (OPNAME.NE.doICYB) NQ = 12*ITE + KADDR
      INFF(22,M) = 2
      GO TO 325
310   ICD(2) = BDAT(M,2)
      IF (ICD(2).NE.doINOSV) GO TO 340
      ICD(1) = BDAT(M,1)
      LBC = LQ
      INFF(16,M) = LBC
      CALL STDAT(M,2,1,2,ICD)
      NQ = 0

```



```

      LFILE = LBC + NBC - 1
      GO TO 330
C  KIND IS SCALAR
315  INFF(20,M) = KND
      NQ = 1
      TYPE = INFF(21,NCD)
      INFF(21,M) = TYPE
      IF (TYPE.NE.doISNGL) NQ = 2
      GO TO 326
C  SET KIND TO A 7X7
320  INFF(20,M) = doIR7
      ITE = 1
C  IF ELEMENT UNDEFINED, LEAVE IT TYPE SINGLE
      IF(NCD.EQ.0) GO TO 324
      ITYP = INFF(21,NCD)
      INFF(21,M) = ITYP
      IF (ITYP.NE.doISNGL) ITE=2
324  IF(OPNAME.NE.doICYB) NQ=49*ITE+KADD
      INFF(22,M)=7
325  INFF(23,M)=7
326  LQ = LFILE + 1
      LFILE = LQ + NQ - 1
      INFF(10,M) = LQ
330  INFF( 11,M) = NQ
      NTOT = NQ + NFL + NIN + NBC
      INFF(18,M) = NTOT
340  CONTINUE
      RETURN
      END

C*****LOC*****
      INTEGER FUNCTION LOC(NAME,NUM,NL,LIST,LOCA,NLOC)
      IMPLICIT REAL*8 (A-H,O-Z)

C
C  SET LOC EQUAL TO REPETITION NUM OF NAME IN LIST, WHERE LIST
C  IS AN ARRAY OF M-NUMBERS OF SYNCH INSTRUCTIONS.
C  MORE PRECISELY, NAME=MNAME(LIST(LOC)).
C  LIST CONTAINS NL M-NUMBERS.
C  ALL OCCURRENCES OF NAME ARE PUT IN ARRAY LOCA,
C  THERE ARE NLOC OF THEM.
C  EXAMPLE —
C  BL   BML           A   B   C   A   D   E   A
C  THEN MI IS THE MI LIST SET UP BY BML THAT CONTAINS THE M'S
C  CORRESPONDING TO THE BEAMLINE A,B,..., WHICH HAS NELS=7 VALUES, AND
C      LOC(A,2,NELS,MI,LOCA,NLOC) = 4
C      LOCA = 1,4,7
C      NLOC = 3
C
      DIMENSION LIST(1),LOCA(1)
      DATA MAX/100/

C
      DO 1 K=1,MAX
1    LOCA(K) = 0
C
      J = 0

```

```

DO 2 I=1,NL
LL = LIST(I)
NAMEI = MNAME(LL)
IF (NAME.NE.NAMEI) GO TO 2
J = J + 1
LOCA(J) = I
IF (J.EQ.MAX) GO TO 3
2 CONTINUE
C
3 NLOC = J
LOC = LOCA(NUM)
RETURN
END
C***** LOCC *****
INTEGER FUNCTION LOCC(NAME,NUM,NL,LIST,LOCA,NLOC)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION LIST(1),LOCA(1)
DATA MAX/100/
C
DO 1 K=1,MAX
1 LOCA(K) = 0
C
J = 0
DO 2 I=1,NL
LL = LIST(I)
NAMEI = MNAME(LL)
IF (NAME.NE.NAMEI) GO TO 2
J = J + 1
LOCA(J) = I
IF (J.EQ.MAX) GO TO 3
2 CONTINUE
C
3 NLOC = J
LOCC = LOCA(NUM)
RETURN
END
C*****LOCS*****
SUBROUTINE LOCS(NAMES,NUMT,NL,LIST,ILIST)
IMPLICIT REAL*8 (A-H,O-Z)
C
DIMENSION NAMES(1),LIST(1),ILIST(1),LOCA(100)
DATA IBLNK1/1H /
C
DO 20 I=1,NUMT
DECODE (4,200,NAMES(I)) NPS1
200 FORMAT (A1,3X)

IF (NPS1.EQ.IBLNK1) GO TO 10
DECODE (4,103,NAMES(I)) NAMEI,NUMI
103 FORMAT(A3,I1)

IF (NUMI.EQ.0) NUMI = 1
ILIST(I) = LOC(NAMEI,NUMI,NL,LIST,LOCA,NLOC)

```

```

      GO TO 20
10  DECODE (4,100,NAMES(I)) ILST

```

```

      ILIST(I) = ILST
100  FORMAT(I4)
20  CONTINUE

```

```

      RETURN
      END

```

```

C ***** MAGMAT *****
      SUBROUTINE MAGMAT(M,KA)

```

```

C          COMPUTE KA NUMBER OF MAGNET MATRICES.
      IMPLICIT REAL*8 (A-H,O-Z)
C  INPUT PARAMETERS MAY BE FL. PT. NUMBERS, SYMBOLIC FL. PT. OR
C  VECTOR NAME FROM WHICH ELEMENT VALUES ARE USED.
      INCLUDE 'CONTRL.CCC'
      DIMENSION LIST(25),ITYP(6),IP(6),PAR(6),IPAR(6)
      EQUIVALENCE (PAR(1),IPAR(1))

```

```

      CALL DATA (M,6,1,6,IP)

```

```

C          DETERMINE TYPE OF PARAMETER.
      DO 10 I=1,6
C  CALL PARTY(P(I),ITYP(I))
      IF (ERROR) RETURN
10  CONTINUE

```

```

C          STORE MAGNET INPUT
C          COMPUTE AND STORE MAGNET MATRICES

```

```

      M1 = M - 1
      DO 25 K=1,KA
      DO 20 I=1,6
      IF (ITYP(I).EQ.3) GO TO 15
C  IP(I) IS FL. PT. NUMBER
      IPAR(I) = IP(I)
      GO TO 20
C  FETCH VECTOR ELEMENT. IP(I) CONTAINS M OF VECTOR.
15  MV = IP(I)
      PAR(I) = FDATA(MV,K)
20  CONTINUE
C  STORE PARAMETERS FOR M1 AND COMPUTE MAGNET MATRICES
      CALL STDAT(M1,1,1,6,PAR)
      CALL MAGNET(M1,PAR)
      M1 = M1 - 1
25  CONTINUE
      RETURN
      END

```

```

C*****MAGNET*****
      SUBROUTINE MAGNET(M,PAR)
C  SET UP MAGNET MATRICES.
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

REAL*8 L,K,KP
INCLUDE 'SWTCH.CCC'
INCLUDE 'CONST.CCC'
DIMENSION RX(2,3),RY(2,3),F(2,3),D(2,3),A(2,3),PAR(1),RW(3)
EQUIVALENCE (C,CH),(S,SH),(RW(1),L),(RW(2),THETA),
1          (RW(3),PL),
2          (F(1,1),D(1,1),A(1,1))
EQUIVALENCE (ALPHA,E1),(BETA,E2),(ANG1,IANG1),(ANG2,IANG2)
INTEGER IDOL/1H$/
DATA IVERT/0/

```

C IN MISALIGNMENT MODE IF BEND = 0.

IF (PAR(1).GE.0) GO TO 8

IVERT = 1

PAR(1) = -PAR(1)

8 L = PAR(1)

PROF = PAR(2)

RHO = PAR(3)

OMEGA = PAR(4)

RHOI = 1.D0/RHO

C CURV = OMEGA/RHO = CURVATURE OF REFERENCE ORBIT

C OMEGA IS POSITIVE IF CENTER OF CURVATURE HAS A NEGATIVE X VALUE

C RELATIVE TO REFERENCE ORBIT.

C

CURV = OMEGA*RHOI

C GAMMA, THE MAGNET WEDGE ANGLE, IS POSITIVE IF THE MAGNET FACES

C INTERSECT ON THE SIDE OF THE REFERENCE ORBIT OPPOSITE TO THE CENTER

C OF CURVATURE.

C

C THETA = L*OMEGA/RHO

THETA = L*CURV

IF (CURV.EQ.0.) GO TO 100

* PAR(5) AND (6) ARE ENTRANCE AND EXIT ANGLES IN DEGREES.

* IF EITHER OR BOTH ARE \$ SIGN, MAGNETS ARE RECTANGULAR.

ANG1 = PAR(5)

ANG2 = PAR(6)

IF ((IANG1.EQ.IDOL).OR.(IANG2.EQ.IDOL)) GO TO 4

E1 = PAR(5) * RADEG

E2 = PAR(6)*RADEG

GO TO 40

* E1,E2 (EQUIVALENCED TO ALPHA AND BETA) ARE ANGLES IN RADIAN

* RECTANGULAR MAGNET

4 IF (PROF.EQ.0..AND.L.NE.0.) GO TO 200

E1 = DABS(THETA)/2.D0

E2 = E1

40 CONTINUE

EFIN = DSIN(ALPHA)*DABS(CURV)/DCOS(ALPHA)

EFOUT = DSIN(BETA)*DABS(CURV)/DCOS(BETA)

ID = 1

FC = 1.D0

C KP = K + OMEGA/RHO

1 KP = FC * PROF + CURV * OMEGA

IF (KP.EQ.0.) GO TO 7

K = DSQRT(DABS(KP*RHOI))

PHI = K*L

```

      IF (KP.LT.0.) GO TO 3
C   FOCUSING MATRIX
2    S = DSIN(PHI)
      C = DCOS(PHI)
      F(1,1) = C
      F(1,2) = S/K
      F(1,3) = (CURV*(1.-C)/(K*K))*BEND
      F(2,1) = -K*S
      F(2,2) = C
      F(2,3) = (CURV*S/K)*BEND
      IF (ID.EQ.1) PL = -(PHI-S)*CURV*CURV/(K*K*K)
      GO TO 10
C   DEFOCUSING MATRIX
3    E = DEXP(PHI)
      EI = 1.D0/E
      CH = 0.5D0*(E+EI)
      SH = 0.5D0*(E-EI)
      D(1,1) = CH
      D(1,2) = SH/K
      D(2,1) = SH*K
      D(1,3) = (CURV*(CH-1.D0)/(K*K))*BEND
      D(2,2) = CH
      D(2,3) = (CURV*SH/K)*BEND
      IF (ID.EQ.1) PL = -(SH-PHI)*CURV*CURV/(K*K*K)
      GO TO 10
C   KP = 0 CASE
7    A(1,1) = 1.D0
      A(1,2) = L
      A(1,3) = (0.5D0*CURV*L*L)*BEND
      A(2,1) = 0.
      A(2,2) = 1.D0
      A(2,3) = (CURV*L)*BEND
      IF (ID.EQ.1) PL = -L*L*L*CURV*CURV/6.D0
C   CORRECT FOR EDGE FOCUSING.
10   A(1,1)=A(1,1)+A(1,2)*EFIN
      A(2,1) = A(1,1)*EFOUT + A(2,1) + A(2,2)*EFIN
      A(2,2) = A(1,2)*EFOUT + A(2,2)
      A(2,3) = A(1,3)*EFOUT + A(2,3)
      IF (ID.EQ.2) GO TO 20
C   SET UP RADIAL TRANSFER MATRIX.
      DO 15 I=1,2
      DO 15 J=1,3
      RX(I,J)=A(I,J)
15   CONTINUE
      IF (IVERT.EQ.0) GO TO 16
      IVERT = 0
      GO TO 26
16   CURV = 0.
      ID = 2
      EFIN=-EFIN
      EFOUT=-EFOUT
      FC=-1.
      GO TO 1
C   SET UP VERTICAL TRANSFER MATRIX.
20   DO 25 I=1,2

```

```

      DO 25 J=1,3
      RY(I,J)=A(I,J)
25    CONTINUE
26    CALL STXY(M,RX,RY,RW)
      RETURN
C QUADRUPOLE
100   PHSQ=PROF*L**2/RHO
      PHI = DSQRT(DABS(PHSQ))
      IF(PHSQ.NE.0.) GO TO 101
      CALL DRIFT(M,L)
      RETURN
101   C= DCOS(PHI)
      S = DSIN(PHI)
      PL=0.
      RX(1,1)=C
      RX(2,2)=C
      RX(1,2)=L*S/PHI
      RX(2,1) = -S*PHI/L
      RX(1,3)=0.
      RX(2,3)=0.
      E = DEXP(PHI)
      CH=(E +1.D0/E)/2.D0
      SH= E - CH
      RY(1,1) = CH
      RY(2,2) = CH
      RY(1,2)=L*SH/PHI
      RY(2,1)=SH*PHI/L
      RY(1,3)=0.
      RY(2,3)=0.
      IF(PHSQ.GT.0.) CALL STXY(M,RX,RY,RW)
      IF(PHSQ.LT.0.) CALL STXY(M,RY,RX,RW)
      RETURN
C RECTANGULAR ZERO-GRADIENT DIPOLE
200   C = DCOS(THETA)
      S = DSIN(THETA)/THETA
      T = THETA*DTAN(THETA/2.D0)
      RX(1,1)=1.D0
      RX(2,2)=1.D0
      RX(1,2)=L*S
      RX(2,1) = 0.
      RX(1,3)=L*(1.D0-C)*BEND/THETA
      RX(2,3)=2.D0*T*BEND/THETA
      PL=RX(1,2)-L

      RY(1,1)=1.D0-T
      RY(2,2)=1.D0-T
      RY(1,2)=L
      RY(2,1)=T*(T-2.D0)/L
      RY(1,3)=0.
      RY(2,3)=0.
      GO TO 26
      END
C ***** MAGRSV *****
      SUBROUTINE MAGRSV( M0,NAME0,KA0,KB0,NAME1 )
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

DATA MXPMY          /1      /
INTEGER NEGZER/-32767/, doIMAG /'MAG '/
1 CONTINUE
CALL STINFO( M0,NAME0,doIMAG,KA0,KB0,NAME1)
CALL RESRV8(M0,MXPMY,NEGZER,6,1,0,0,0)

C
RETURN
END

C ***** MAGS *****
SUBROUTINE MAGS(M,N,Z)
IMPLICIT REAL*8 (A-H,O-Z)
REAL*4 ran6a
INCLUDE 'BINFF.CCC'
INCLUDE 'SWTCH.CCC'
INCLUDE 'DIM.CCC'
LOGICAL BSW
DIMENSION RX(2,3),RY(2,3),RW(3),Z(5),Y(6),X(4),ML(3)
DIMENSION ZZ(5)
INCLUDE 'MI.CCC'
common/seed/nr
DATA RW /3*0./
EQUIVALENCE (Y(2),X(1),ML(1))

C COMPUTE MATRIX WITH MISALIGNMENTS INCLUDED
C RETRIEVE LENGTH
C RETRIEVE MATRIX ELEMENTS OF INDEX N AND STORE IN M.
C EQUATES M TO N INITIALLY. N WILL BE PRESERVED.A NEW MATRIX WILL BE
C COMPUTED FOR M
C MS AND MT ARE WORKING STORAGE LOCATIONS

MS=MEND-1
MT=MEND-2
KA = INFF(4,M)
IF (KA.EQ.0) KA = 1
C IF(KB.GE.1) ERRORS ARE RANDOM WITH GIVEN RANGE
KB=INFF(5,M)
DO21 I=1,5
21 ZZ(I)=Z(I)
IF(KB.LE.0) GO TO 20
DO22 I=1,5
22 Z(I)=ZZ(I)*(ran(nr)-.5)
20 CONTINUE
DBB = Z(5)
BSW = .FALSE.
IF(Z(5).EQ.0) GO TO 10
C GET MAGNET PARAMETERS
2 CALL DATA(N,1,1,6,Y)
IF (KA.EQ.2) GO TO 3
C INPUT DB/B. ALTER RHO
Y(3) = Y(3)/(1.D0+Z(5))
GO TO 7
C INPUT DB. TEST FOR B=0. INCREMENT B
3 IF (Y(4).EQ.0.) BSW = .TRUE.
IF (.NOT.BSW) DBB = Z(5)/Y(4)
Y(4) = Y(4) + Z(5)

```

```

7   SMS = BEND
   BEND=1.D0
C   COMPUTE MAGNET MATRIX WITH FIELD ERROR INCLUDED
   CALL MAGNET(M,Y)
   CALL RXY(M,RX,RY,RW)
   IF (BSW) GO TO 8
C   WHERE B NOT 0.
   RX(1,3) = - RX(1,3)*DBB
   RX(2,3) = -RX(2,3)*DBB
   GO TO 9
C   WHERE B = 0.
8   CURV = Y(4)/Y(3)
   EL = Y(1)
   XX = -(EL*EL/2.D0)*CURV
   XP = -EL*CURV
   RX(1,3) = XX
   RX(2,3) = XP
9   CALL STXY(M,RX,RY,RW)
C   RESTORE MISALIGNMENT MODE SWITCH
   BEND=SMS
4   Y(6)=0.
   DO 5 I=1,4
5   Y(6) = Y(6) + DABS(Z(I))
   IF (Y(6).EQ.0.0) GO TO 6
C   COMPUTE POST SHIFT MATRIX
   X(1)=Z(2)
   X(2)=(Z(2)-Z(1))/Y(1)
   X(3)=Z(4)
   X(4)=(Z(4)-Z(3))/Y(1)
   CALL SHF(MS,X)
C   COMPUTE PRE SHIFT MATRIX
   X(1)=-Z(1)
   X(2)=-X(2)
   X(3)=-Z(3)
   X(4)=-X(4)
   CALL SHF(MT,X)
   ML(1)=MT
   ML(2)=M
   ML(3)=MS
   MLL(1) = ML(1)
   MLL(2) = ML(2)
   MLL(3) = ML(3)
   CALL MMM(M,3,MLL)
6   CONTINUE
DO23 I=1,5
23  Z(I)=ZZ(I)
   RETURN
10  Y(1)=FDAT(N,1)
   CALL RXY(N,RX,RY,RW)
   CALL STXY(M,RX,RY,RW)
   GO TO 4
   END
C ***** MAGSRV *****
   SUBROUTINE MAGSRV( M0,NAME0,KAO,KB0,NAME1 )
   IMPLICIT REAL*8 (A-H,O-Z)

```



```

DATA MXPMY          /1      /
INTEGER NEGZER/-32767/
CALL STINFO( M0,NAME0,4HMAGS ,KA0,KB0,NAME1)
CALL INP8(M0,MXPMY,NEGZER,1,2,5,1,0)
RETURN
END

C ***** MAGV *****
SUBROUTINE MAGV(M,PAR)
IMPLICIT REAL*8 (A-H,O-Z)
C          SET UP MATRIX WITH VERTICAL BEND
C  IF B IS +, THEN DO +PI/2 M -PI/2
C  IF B IS -, THEN DO -PI/2 M +PI/2

INCLUDE 'DIM.CCC'
INCLUDE 'CONST.CCC'
DIMENSION T(49),TR(49),RW(3),PAR(1)

C  COMPUTE MATRIX FOR BENDING MAGNET
CALL MAGNET(MEND,PAR)
CALL RTRV7(MEND,T,RW)
THETA = PIO2
CALL ROTATE (THETA,T,TR,RW)
CALL STOR7(M,TR,RW)
RETURN
END

C ***** MAP *****
SUBROUTINE MAP(V,PAR)
RETURN
END

C ***** MAP3 *****
SUBROUTINE MAP3(V,PAR)
IMPLICIT REAL*8 (A-H,O-Z)

C          SIMULATE BEAM-BEAM FORCE WITH ROUND BEAM
C  PAR(1) = -DNUY    PAR(2) = RADIUS,  PAR(3) = BETA-Y

DIMENSION V(4),PAR(3)
INCLUDE 'CONST.CCC'

RS = V(3)**2/PAR(2)**2
IF (V(3).EQ.0.) GO TO 2
F = 2.*PI2*PAR(1)*(1.-DEXP(-RS))/(RS*PAR(3))
3  V(4) = V(4) + V(3)*F
2  RETURN
END

C ***** MAP8 *****
SUBROUTINE MAP8(V,PAR)
RETURN
END

C ***** MAP9 *****
SUBROUTINE MAP9(V,PAR)
RETURN
END

C ***** MATOUT *****
SUBROUTINE MATOUT(TRACE,KODE)

```

IMPLICIT REAL*8 (A-H,O-Z)

CC PRINTS THE COVARIANCE MATRIX V. CALCULATES AND PRINTS THE
CC INDIVIDUAL CORRELATION COEFFICIENTS AND GLOBAL CORRELATIONS.
CC

```
COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
INCLUDE 'PAREXT.CCC'
COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
COMMON /VARIAN/ V(15,15)
COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
COMMON /SIMVEC/ P(15,16) ,PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15)
DIMENSION VLINE(15)
IF (ISW(2) .LT. 2) RETURN
WRITE (ISYSWR,600)
IF (TRACE .NE. 0.0D0) WRITE (ISYSWR,610) TRACE
IF (NPAR .EQ. 0) GO TO 250
ISWTR = ISW(5) - ITAUR
IF (ISWTR .LT. 2) GO TO 120
```

C INTERNAL COVARIANCE MATRIX

```
DO 100 I= 1, NPAR
100 WRITE (ISYSWR,620) (V(I,J), J=1,I)
WRITE (ISYSWR,630)
120 CONTINUE
```

C CORRELATION COEFFS. .

```
IF (KODE .LT. 1) GO TO 500
IF (NPAR .LE. 1) GO TO 500
WRITE (ISYSWR, 650)
NPARM = MIN0(NPAR-1, 18)
WRITE (ISYSWR,690) (ID ID=1,NPARM)
DO 200 I= 2, NPAR
IM = I-1
DO 170 J= 1, IM
170 VLINE(J) = V(I,J)/DSQRT(DABS(V(I,I)*V(J,J)))
200 WRITE (ISYSWR,660) I,(VLINE(IZ),IZ=1,IM)
250 CONTINUE
WRITE (ISYSWR, 630)
```

C GLOBAL CORRELATION COEFFS . . .

```
CALL UCOPY(V,P,MAXINT**2)
CALL VERMIN(P,MAXINT,MAXINT,NPAR,IERR)
IF (IERR .GT. 0) RETURN
WRITE (ISYSWR,670)
DO 400 I= 1, NU
L = LCORSP(I)
IF (L .EQ. 0) GO TO 400
GCC = 1.0D0 - 1.0D0/(V(L,L)*P(L,L))
WRITE(ISYSWR,680) I, NAM(I), GCC
400 CONTINUE
500 RETURN
600 FORMAT(/11X 26HINTERNAL COVARIANCE MATRIX)
610 FORMAT (1H+43X 28H LAST FRACTIONAL CHANGE WAS F10.6)
620 FORMAT (1H 9X11E10.3)
630 FORMAT (1H )
650 FORMAT (11X 24HCORRELATION COEFFICIENTS)
660 FORMAT (8X,13,18F6.3/(20F6.3))
670 FORMAT (64X 18HGLOBAL CORRELATION/ 54X 25HPARAMETER COEFFI
```

```

1CIENT)
680 FORMAT (47X,15,2X,A10,F13.5)
690 FORMAT (8X,4HINT.,14,1716)
END
C ***** MAXMIN *****
SUBROUTINE MAXMIN(BMAX,BMIN,BET,MB,IP)
  IMPLICIT REAL*8 (A-H,O-Z)

C   DETERMINES ABSOLUTE MAXIMUM VALUE —BMAX OR BET
C   DETERMINES ABSOLUTE MINIMUM VALUE —BMIN OR BET
C   IF TWO ARE EQUAL, THE FIRST REMAINS AS MIN OR MAX
C   MB(1),MB(2) ARE INDICES OF MAX. AND MIN. VALUES
C   IP = INDEX OF BET

  DIMENSION MB(1)
  IF (DABS(BMAX).GT.DABS(BET)) GO TO 10
  BMAX = BET
  MB(1) = IP
10  IF (DABS(BMIN).LT.DABS(BET)) RETURN
  BMIN = BET
  MB(2) = IP
  RETURN
  END

C ***** MDAT *****
FUNCTION MDAT(M,J)
  IMPLICIT REAL*8 (A-H,O-Z)
C   IF THE JTH BCD STORAGE LOCATION OF M CONTAINS AN ELEMENT NAME,
C   THEN MDAT WILL CONTAIN THE M-NUMBER( OR INDEX ) OF THAT ELEMENT.
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))

  INTEGER ELNUM
  LOCI= INFF(16,M) * 2-1
  LOC=LOCI+J-1
  MDAT = ELNUM(ISTORE( LOC))
  RETURN
  END

C ***** MESH *****
SUBROUTINE MESH(M,KA,MOP)
  IMPLICIT REAL*8 (A-H,O-Z)
  DIMENSION IND(10,2),NM(10,2),FMIN(10,2),FDEL(10,2),VSAV(10,2),
1  LP(10,2),LL(10,2),JN(10,2),FPAR(10,2)
  DIMENSION FMAX(10),KEJ(2),F(30)
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  INCLUDE 'SWTCH.CCC'
  INCLUDE 'CONTRL.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))
  INTEGER do1VPAR/'VPAR'/
  INTEGER ELNUM
  JX = 1

```

```

      IF (MOP.EQ.doIVPAR) JX = 2
      IF (.NOT.MSSW) GO TO 5
      IF (JX.GT.1) GO TO 4
      IF (MHS) GO TO 46
      GO TO 5
4     IF (VPR) GO TO 48
C     INITIAL CALL
C     RETRIEVE VARIABLE NAMES AND INDEX NUMBER
5     CONTINUE
      CALL DATA(M,2,2,KA,NM(1,JX) )
      CALL DATA(M,3,1,KA,IND(1,JX) )
C     RETRIEVE MESH LIMITS THAT WERE INPUT
      KM = KA*3
      CALL DATA(M,1,1,KM,F)
      I1=1
      I2=2
      I3=3
C
      DO 10 I=1,KA
      FMIN(I,JX) = F(I1)
      FMAX(I)=F(I2)
      FDEL(I,JX) = F(I3)
      I1=I1+3
      I2 = I2 + 3
      I3 = I3+3
10    CONTINUE
C     COMPUTE LOOP LIMITS,SAVE ORIGINAL VALUES,SET BEGINNING VALUES OF MESH
      KEJ(JX) = KA
      KE = KA
      DO 15 I=1,KE
      LL(I,JX) = ((FMAX(I)-FMIN(I,JX))/FDEL(I,JX)) + 1.0000001D0
      JN(I,JX) = ELNUM(NM(I,JX))
      F(I) = FDAT(JN(I,JX),IND(I,JX))
      VSAV(I,JX) = F(I)
      FPAR(I,JX) = FMIN(I,JX)
      CALL STDAT(JN(I,JX),1,IND(I,JX),1,FPAR(I,JX) )
15    CONTINUE
C     INITIALIZE LOOP INDICES.  RUNNING INDEX OF LOOP IS LP.  LIMIT IS LL
      DO 20 I=1,KE
      LP(I,JX) = 1
20    CONTINUE
      IF (JX.EQ.2) GO TO 30
      MHS=.TRUE.
      GO TO 60
30    LMX = LL(1,JX)
      DO 31 I=2,KE
      IF (LL(I,JX).GT.LMX) LMX = LL(I,JX)
31    CONTINUE
      LMM = 1
      VPR = .TRUE.
      GO TO 60
C     VPAR CALCULATION.  STRAIGHT LINE
48    CONTINUE
C     TEST TO SEE IF FINISHED
      KE = KEJ(JX)

```

```

      IF(LMM.EQ.LMX) GO TO 63
      LMM = LMM + 1
      DO 36 I=1,KE
      FPAR(I,JX) = FPAR(I,JX) + FDEL(I,JX)
      CALL STDAT(JN(I,JX),1,IND(I,JX),1,FPAR(I,JX))
36    CONTINUE
      GO TO 60
C  FULL MESH
46    CONTINUE
      KE = KEJ(JX)
      KV = KE
      I = 1
49    IF (LP(KV,JX).NE.LL(KV,JX))GO TO 55
C  LOOP COMPLETED.  RESET
      LP(KV,JX) = 1
      FPAR(KV,JX) = FMIN(KV,JX)
      CALL STDAT(JN(KV,JX),1,IND(KV,JX),1,FPAR(KV,JX))
      KV = KV - 1
      IF (I.EQ.KE) GO TO 64
      I = I + 1
      GO TO 49
55    LP(KV,JX) = LP(KV,JX) + 1
      FPAR(KV,JX) = FPAR(KV,JX) + FDEL(KV,JX)
      CALL STDAT(JN(KV,JX),1,IND(KV,JX),1,FPAR(KV,JX) )
60    KA = 1
      WRITE (3,100)
      DO 62 I=1,KE
      WRITE (3,101) IND(I,JX),A(I,JX),FPAR(I,JX)
62    CONTINUE
      CALL SUBR(3)
      MSSW = .TRUE.
      MODS=.TRUE.
      RETURN
C  FINISHED.  REPLACE ORIGINAL VALUES OF VARIABLES
63    VPR=.FALSE.
      MODS=.FALSE.
      GO TO 65
64    MHS=.FALSE.
      MODS=.FALSE.
65    DO 70 I=1,KE
      F(I) = VSAV(I,JX)
      CALL STDAT(JN(I,JX),1,IND(I,JX),1,F(I))
70    CONTINUE
      IF(MHS) RETURN
      IF (VPR) RETURN
      MSSW = .FALSE.
      MODE = INFF(19,M)
      RETURN
100   FORMAT (/15H CURRENT VALUES/)
101   FORMAT (' ', 5X,I5,4H OF ,A5,3H = ,F10.5)
      END
C ***** MFD *****
      SUBROUTINE MFD(M,J,FACTOR)
      IMPLICIT REAL*8 (A-H,O-Z)
C  MULTIPLIES BY FACTOR THE J-TH FL PT PARAMETER OF ELEMENT M AND

```

C RETURNS NEW VALUE TO STORAGE.

```

F=FDAT(M,J)
F=F * FACTOR
CALL REPFLT(M,J,F)
RETURN
END

```

C*****MICADO*****

```

SUBROUTINE MICADO(A,B,NDIM,M,N,AP,XA,NA,NB,NC,EPS,ITER,DP,X,NX,R,
1RHO)

```

C*****

```

IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(NDIM,1),B(1),AP(1),XA(1),NA(1),NB(1),NC(1),X(1),NX(1)
1,R(1),RHO(1)

```

C

C REDUCTION DE AP ET DE A

```

CALL NBR(NA,N1)
CALL NBR(NC,N3)
CALL NBR(NB,N2)
M1=M-N3
M2=N-N2

```

C CALL MATRED(A,NDIM,N,N3,N2,NC,NB)

C CALL MATRED(AP,1,M,0,N3,0,NC)

C

C***PRISE EN COMPTE DES CONDITIONS INITIALES

C

```

IF(N1.EQ.0)GO TO 4
DO 1 I=1,N1
K=0
IF(N2.EQ.0)GO TO 1
DO 2 J=1,N2
IF(NA(I).GT.NB(J))K=K+1
2 CONTINUE
1 NA(I)=NA(I)-K
DO 3 I=1,M1
DO 3 J=1,N1
K=NA(J)
3 B(I)=B(I)-A(I,K)*XA(J)

```

C

C CALCUL DE DP/P

C

```

4 S1=0.0
S2=0.0
DO 5 I=1,M1
S1=S1+B(I)*AP(I)
5 S2=S2+AP(I)*AP(I)
IF(S2.EQ.0.0)GO TO 6
DP=S1/S2
DO 21 I=1,M1
21 B(I)=B(I)-DP*AP(I)
6 NN=N+N
DO 7 I=1,M1
RHO(NN+1)=B(I)
7 B(I)=-B(I)
CALL HTLS(A,B,NDIM,M1,M2,ITER,X,NX,R,RHO,EPS)

```

```

      DO 8 I=1,M
      R(I)=-R(I)
8    B(I)=RHO(NN+I)
      IF(N2.EQ.0) RETURN
      DO 9 J=1,N2
      DO 9 I=1,M2
9    IF(NX(I).GE.NB(J)) NX(I)=NX(I)+1
      RETURN
      END
C ***** MIDATA *****
      SUBROUTINE MIDATA
      IMPLICIT REAL*8 (A-H,O-Z)
CC      READS THE DATA CARDS (TITLE CARD AND PARAMETER CARDS)
CC      AND SETS UP THE STARTING PARAMETER LISTS.
CC      CONTROL THEN PASSES TO COMAND FOR READING THE COMMAND CARDS.
CC
      COMMON /MINERR/ ERP(30) ,ERN(30)
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON /MATCH2/ AA(30),BB(30),KBTS,SMINCL,
1      DWORD(10),DWORD2(10),DWORD3(10),DWORD7(7,10),MINCOM
      LOGICAL SMINCL
      INTEGER DWORD,DWORD2,DWORD3
C      . . . INITIALIZE NEW DATA BLOCK . . .
      NBLOCK = NBLOCK + 1
      VERSN = 1.75D0
      WRITE (ISYSWR,1004) VERSN,NBLOCK
      WRITE (ISYSWR,1005)
      DO 50 I= 1, 7
50    ISW(I) = 0
      SIGMA = 0.D0
      WRITE (ISYSWR,1005)
      NPFIX = 0
      NINT = 0
      NU = 0
      NPAR = 0
      IFATAL = 0
      WRITE (ISYSWR,1005)
      DO 100 I= 1, MAXEXT
      ERP(I) = 0.0D0
      ERN(I) = 0.0D0
      LCODE(I) = 0
100    LCORSP (I) = 0
      UP = 1.0D0
      ISW(5) = 1
C      . . . READ PARAMETER CARDS . . .
      DO 200 K=1,KBTS
      NAMK=NAM(K)
      UK=U(K)
      WK=WERR(K)
      A=AA(K)

```

```

      B=BB(K)
      NU = MAX0(NU,K)
      IF (K .LE. MAXEXT) GO TO 115
      IFATAL = IFATAL + 1
      WRITE (ISYSWR,1009) K,MAXEXT
      WRITE (ISYSWR,1002) K,NAMK,UK,WK,A,B
      GO TO 160
115 CONTINUE
      IF (WK .GT. 0.0D0) GO TO 122
C      . . . FIXED PARAMETER . . . . .
      WRITE (ISYSWR, 1002) K,NAMK,UK
      LCODE(K) = 0
      GO TO 160
C      . . . VARIABLE PARAMETER . . .
122 WRITE (ISYSWR, 1002) K,NAMK,UK,WK,A,B
      NINT = NINT + 1
      IF (A) 140,130,140
130 IF (B) 140,135,140
135 LCODE(K) = 1
      GO TO 160
140 IF (B-A) 145,142,150
142 IFATAL = IFATAL + 1
      WRITE (ISYSWR,1010)
      GO TO 150
145 SAV = B
      B = A
      A = SAV
150 ALIM(K) = A
      BLIM(K) = B
      LCODE(K) = 4
      IF ((B-U(K))*(U(K)-A)) 153,155,160
153 IFATAL = IFATAL + 1
      WRITE (ISYSWR,1011)
      GO TO 160
155 WRITE (ISYSWR,1006)
160 CONTINUE
200 CONTINUE
C      . . . END PARAMETER CARDS
C      . . . STOP IF FATAL ERROR
250 WRITE (ISYSWR,1005)
      IF (NINT .LE. MAXINT) GO TO 253
      WRITE (ISYSWR,1008) NINT,MAXINT
      IFATAL = IFATAL + 1
253 IF (IFATAL .LE. 0) GO TO 280
      WRITE (ISYSWR,1013) IFATAL
      STOP
C      CALCULATE STEP SIZES DIRIN
280 NPAR = 0
      DO 300 K= 1, NU
      IF (LCODE(K) .LE. 0) GO TO 300
      NPAR = NPAR + 1
      LCORSP(K) = NPAR
      SAV = U(K)
      X(NPAR) = PINTF(SAV,K)
      XT(NPAR) = X(NPAR)

```



```

    SAV2 = SAV + WERR(K)
    VPLU = PINTF(SAV2,K) - X(NPAR)
    SAV2 = SAV - WERR(K)
    VMINU = PINTF(SAV2,K) - X(NPAR)
    DIRIN(NPAR) = 0.5D0 * (DABS(VPLU) + DABS(VMINU))
300 CONTINUE
    RETURN
1110 FORMAT (12X, 15A4, 25X 4HTIME,F7.3)
C...   THE FORMAT BELOW IS MACHINE-DEPENDENT. (A10) , (A6,4X) , ETC.
1001 FORMAT (F10.0, A10, 4F10.0)
1002 FORMAT (10X110,5X,A10,2F15.6,5X,2E15.4)
1003 FORMAT ( 53H WARNING                - ABOVE LIMITS HAVE BEEN REVERSED.
1)
1004 FORMAT (1H1/50X 21(1H*)/50X
1      21H*   D506   MINUTS   *
2/50X11H*   VERSION F6.2,4H   */50X16H* DATA BLOCK NO. 13,2H *)
1005 FORMAT (12X96(1H*))
1006 FORMAT ( 49H WARNING                - ABOVE PARAMETER IS AT LIMIT )
1007 FORMAT ( 106H WARNING ***** - PARAMETER REQUESTED ON FOLLOWING
1 CARD HAS ALREADY APPEARED. PREVIOUS VALUES IGNORED.)
1008 FORMAT ( 46H0   TOO MANY VARIABLE PARAMETERS. YOU REQUEST15/ 49
1H   THIS VERSION OF MINUIT IS ONLY DIMENSIONED FOR14//)
1009 FORMAT ( 30H0FATAL ERROR. PARAMETER NUMBER I11, 29H GREATER THAN
1 ALLOWED MAXIMUM I4)
1010 FORMAT ( 47H FATAL ERROR. UPPER AND LOWER LIMITS ARE EQUAL.)
1011 FORMAT ( 38H FATAL ERROR. PARAMETER OUTSIDE LIMITS/)
1012 FORMAT ( 43H0FATAL ERROR. MORE THAN 200 PARAMETER CARDS /)
1013 FORMAT (/I5, 41H FATAL ERRORS ON PARAMETER CARDS. ABORT.//)
END
C ***** MIEXP *****
SUBROUTINE MIEXP(M,J,NDAT,NELS,ML)
IMPLICIT REAL*8 (A-H,O-Z)

DIMENSION ML(1),KBEG(10),NREP(10),NR(10)
INTEGER SHFTR
INTEGER BK,BDAT
DATA IBLANK/' ' /
DATA MASK/ZFF000000/
C DATA FOR ( AND )
DATA IPL/Z00000028/
DATA IPR/Z00000029/

C EXPAND PARENTHESES AND STORE IN ML. KI INPUT IS NUMBER OF
C 5-CHARACTER BCD DATA TO PROCESS, ON OUTPUT IS NUMBER OF M;S STORED.

```

```

    I=0
    LOOP=0
    K=1
1 IF (K.GT.NDAT) GO TO 5
    BK=BDAT(M,K+J-1)
    IF (BK.EQ.IBLANK) GO TO 6
    LBK=LAND(BK,MASK)
    LBK = SHFTR( LBK, 24)
    IF (LBK.EQ.IPL) GO TO 2
    IF (LBK.EQ.IPR) GO TO 3

```

```

      I=I+1
      ML(I)=MDAT(M,K+J-1)
6     K=K+1
      GO TO 1
2     DECODE(4,1000,BK)NUM
1000 FORMAT (I3,Ix)
      IF(NUM.EQ.0) GO TO 3
      LOOP=LOOP+1
      KBEG(LOOP)=K+1
      NREP(LOOP)=NUM
      NR(LOOP)=0
      K=K+1
      GO TO 1
3     NR(LOOP)=NR(LOOP)+1
      IF(NR(LOOP).EQ.NREP(LOOP)) GO TO 4
      K=KBEG(LOOP)
      GO TO 1
4     K=K+1
      LOOP=LOOP-1
      GO TO 1
5     NELS = I
      RETURN
      END

```

C ***** MIEXPA *****

```

      SUBROUTINE MIEXPA(M,J,NDAT,NELS,ML)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

      INCLUDE 'ALTM1.CCC'

```

C ALTM1.CCC

```

C     PARAMETER (NPMX=150)
C     COMMON/ALTM1/NPN,NP1(NPMX),NP2(NPMX)

```

```

      DIMENSION ML(1),KBEG(10),NREP(10),NR(10)
      INTEGER SHFTR
      INTEGER BK,BDAT
      DATA IBLANK/'  '/
      DATA MASK/ZFF000000/
C     DATA FOR ( AND )
      DATA IPL/Z00000028/
      DATA IPR/Z00000029/

```

C EXPAND PARENTHESES AND STORE IN ML. KI INPUT IS NUMBER OF
C 5-CHARACTER BCD DATA TO PROCESS, ON OUTPUT IS NUMBER OF M;S STORED.

```

      I=0
      LOOP=0
      K=1
1     IF (K.GT.NDAT) GO TO 5
      BK=BDAT(M,K+J-1)
      IF (BK.EQ.IBLANK) GO TO 6
      LBK=LAND(BK,MASK)
      LBK = SHFTR( LBK, 24)
      IF (LBK.EQ.IPL) GO TO 2
      IF (LBK.EQ.IPR) GO TO 3
      I=I+1

```

```

      ML(I)=MDAT(M,K+J-1)
C-----INSERT FOR BMLI SITUATION
      DO 10 IJ = 1,NPN
      IF (ML(I).NE.NP1(IJ)) GO TO 10
      ML(I) = NP2(IJ)
      GO TO 6
10    CONTINUE
C-----
6      K=K+1
      GO TO 1
2      DECODE(4,1000,BK)NUM
1000  FORMAT (I3,1X)
      IF(NUM.EQ.0) GO TO 3
      LOOP=LOOP+1
      KBEG(LOOP)=K+1
      NREP(LOOP)=NUM
      NR(LOOP)=0
      K=K+1
      GO TO 1
3      NR(LOOP)=NR(LOOP)+1
      IF(NR(LOOP).EQ.NREP(LOOP)) GO TO 4
      K=KBEG(LOOP)
      GO TO 1
4      K=K+1
      LOOP=LOOP-1
      GO TO 1
5      NELS = I
      RETURN
      END

```

```

C ***** MIFILL *****
      SUBROUTINE MIFILL(M,J,NDAT,NELS,MI)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

      INCLUDE 'BINFF.CCC'
C BINFF.CCC
C      COMMON /BINFF/ INFF(24,1000)

      INCLUDE 'ALTM1.CCC'
C ALTM1.CCC
C      PARAMETER (NPMX=150)
C      COMMON/ALTM1/NPN,NP1(NPMX),NP2(NPMX)

```

```

      COMMON/ME/ME
      DIMENSION MI(1), ME(16000)
      INTEGER OPL
      LOGICAL FBSW

      FBSW = .FALSE.
      MN = MDAT(M,1)
      NOP = INFF(1,MN)
      IF (NOP.EQ.'BMLI') FBSW = .TRUE.

      IF (FBSW) THEN
        CALL FNBML(MN,NPN,NP1,NP2)

```

```

        CALL MIEXPA(M,J,NDAT,NELS,ME)
    ELSE
        CALL MIEXP(M,J,NDAT,NELS,ME)
    END IF
    KREF=1
    OPL = INFF(1,M)
    IF (OPL.NE.'BML') GO TO 25
    KAL = INFF(4,M)
    IF (KAL.GE.0) GO TO 25
    DO 20 N =1,NELS
20  MI(N) = ME(NELS-N+1)
    KREF = -KREF
    GO TO 27
25  DO 26 N = 1,NELS
26  MI(N) = ME(N)
27  KE = NELS
    L=0

1   L=L+1
    IF(L.GT.KE) GO TO 2
    ML=MI(L)

    IF (ML.GT.999990) GO TO 3
    OPL=INFF(1,ML)
    KAL=INFF(4,ML)

    IF (OPL.NE.'BML'.AND.OPL.NE.'LIST'.AND.OPL.NE.'BMLI') GO TO 1
    IF (OPL.EQ.'BMLI') THEN
        FBSW = .TRUE.
        MLF = MDAT(ML,1)
        CALL FNBML(ML,NPN,NP1,NP2)
        ML = MDAT(MLF,1)
        OPL = INFF(1,ML)
        KAL = INFF(4,ML)
        KL = INFF(17,ML)
        CALL MIEXPA(ML,1,KL,NL,ME)
    ELSE
        KL=INFF(17,ML)
        IF (FBSW) THEN
            CALL MIEXPA(ML,1,KL,NL,ME)
        ELSE
            CALL MIEXP(ML,1,KL,NL,ME)
        END IF
    END IF

    IF (OPL.EQ.'BML'.OR.OPL.EQ.'BMLI') GO TO 11
    KBL=INFF(5,ML)

    IF (KREF.LT.0.AND.KAL.NE.1) GO TO 12
    MI(L)=ME(KBL)
    GO TO 13

12  MI(L)=ME(NL-KBL+1)

13  KBL=KBL+1

```

```

      IF (KBL.LE.NL) GO TO 14
      KBL=1
14      INFF(5,ML)=KBL
      GO TO 1

11      KEL=KE-L
      N0=1

      IF(KAL.GE.0) GO TO 4
      N0=0
      KE=KE+1
      KREF=-KREF

4      DO 5 N=1,KEL
5      MI(KE+NL-N)=MI(KE-N+N0)

      IF(KREF.LT.0) GO TO 7
      DO 6 N=1,NL
6      MI(L+N-1)=ME(N)
      GO TO 9

7      DO 8 N=1,NL
8      MI(L+N-1)=ME(NL-N+1)

9      KE=KE+NL-1
      IF (KAL.LT.0) MI(L+NL)=999999
      L=L-1
      GO TO 1

3      KEL=KE-L
      DO 10 N=1,KEL
10     MI(L+N-1)=MI(L+N)
      KE=KE-1
      L=L-1
      KREF=-KREF
      GO TO 1

2      NELS=KE

      RETURN
      END

```

C MIGRAD

```

      SUBROUTINE MIGRAD
      IMPLICIT REAL*8 (A-H,O-Z)
CC      PERFORMS A LOCAL FUNCTION MINIMIZATION USING BASICALLY THE
CC      METHOD OF DAVIDON-FLETCHER-POWELL AS MODIFIED BY FLETCHER
CC      REF. — FLETCHER, COMP.J. 13,317 (1970) 'SWITCHING METHOD'
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU

```

```

COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
COMMON /DERIVA/ G(30) ,G2(30)
DIMENSION GS(30),R(15),XXS(15),FLNU(15),VG(15),VII(15)
DATA SLAMIN,SLAMAX,TLAMIN,TLAMAX/0.2D0, 3.0D0, 0.05D0, 6.0D0/
IF (NPAR .LE. 0) RETURN
ISWTR = ISW(5) - ITAU
NPFN = NFCN
PARN=NPAR
RHO2 = 10.D0*APSI
ROSTOP = 1.0D-5 * APSI
TRACE=1.0D0
IFLAG=4
IF (ISW(3) .EQ. 1) IFLAG = 2
FS = AMIN
IF (ITAU .LT. 1) WRITE (ISYSWR,470) ROSTOP, APSI, VTEST
GO TO 2
1 WRITE (ISYSWR,520)
C . . . . . STEP SIZES DIRIN . . .
2 NPARD = NPAR
DO 3 I= 1, NPAR
D = 0.02D0* DABS(DIRIN(I))
IF (ISW(2) .GE. 1) D = 0.02D0* DSQRT(DABS(V(I,I))*UP)
IF (D .LT. 1.0D-8 *DABS(X(I))) D = 1.0D-8 * X(I)
3 DIRIN(I) = D
C . . . . . STARTING GRADIENT
NTRY = 0
4 NEGG2 = 0
DO 10 ID= 1, NPARD
I = ID + NPAR - NPARD
D = DIRIN(I)
XTF = X(I)
X(I) = XTF + D
CALL INTOEX(X)
CALL FCN(NPAR,G,FS1,U,4)
NFCN = NFCN + 1
X(I) = XTF - D
CALL INTOEX(X)
CALL FCN(NPAR,G,FS2,U,4)
NFCN = NFCN + 1
X(I) = XTF
GS(I) = (FS1-FS2)/(2.0D0 * D)
G2(I) = (FS1 + FS2 - 2.0D0*AMIN) / D**2
IF (G2(I) .GT. 1.0D-30) GO TO 10
C . . . SEARCH IF G2 .LE. 0. . .
WRITE (ISYSWR,520)
NEGG2 = NEGG2 + 1
NTRY = NTRY + 1
IF (NTRY .GT. 4) GO TO 230
D = 50.D0*DABS(DIRIN(I))
XBEG = XTF
IF (GS(I) .LT. 0.0D0) DIRIN(I) = -DIRIN(I)
KG = 0
NF = 0

```

```

      NS = 0
5  X(I) = XTF + D
      CALL INTOEX(X)
      CALL FCN(NPAR,G,F,U,4)
      NFCN = NFCN + 1
      IF (F .LE. AMIN) GO TO 6
C      FAILURE
      IF (KG .EQ. 1) GO TO 8
      KG = -1
      NF = NF + 1
      D = -0.4D0*D
      IF (NF .LT. 10) GO TO 5
      D = 1000.D0*D
      GO TO 7
C      SUCCESS
6  XTF = X(I)
      D = 3.0D0*D
      AMIN = F
      KG = 1
      NS = NS + 1
      IF (NS .LT. 10) GO TO 5
      IF (AMIN .LT. FS) GO TO 8
      D = 0.001D0*D
7  XTF = XBEG
      G2(I) = 1.0D0
      NEGG2 = NEGG2 - 1
8  X(I) = XTF
      DIRIN(I) = 0.1D0*D
      FS = AMIN
10 CONTINUE
      IF (NEGG2 .GE. 1) GO TO 4
      NTRY = 0
      MATGD = 1
C
      IF (ISW(2) .GT. 1) GO TO 15
11 NTRY = 1
      MATGD = 0
      DO 13 I= 1, NPAR
      DO 12 J= 1, NPAR
12 V(I,J) = 0.0D0
13 V(I,I) = 2.0D0/G2(I)
C
      IF (ISW(2) .GT. 1) GO TO 15
15 SIGMA = 0.0D0
      DO 18 I= 1, NPAR
      IF (V(I,I) .LE. 0.) GO TO 11
      RI = 0.0D0
      DO 17 J= 1, NPAR
      XYS(I) = X(I)
17 RI= RI+ V(I,J) * GS(J)
18 SIGMA = SIGMA + GS(I) *RI *0.5D0
      IF (SIGMA .GE. 0.0D0) GO TO 20
      WRITE (ISYSWR,520)
      IF (NTRY.EQ.0) GO TO 11
      ISW(2) = 0
      GO TO 230

```

. DIAGONAL MATRIX

. . . GET SIGMA AND SET UP LOOP

```

20 ISW(2) = 1
   ITER = 0
   CALL INTOEX(X)
   IF (ISWTR .GE. 1) CALL MPRINT(0,AMIN)
   IF (ISWTR .GE. 2) CALL MATOUT(0.0D0, 1)
C                                     . . . . . START MAIN LOOP
24 CONTINUE
   GDEL = 0.0D0
   DO 30 I=1,NPAR
     RI = 0.0D0
     DO 25 J=1,NPAR
25 RI = RI + V(I,J) *GS(J)
     DIRIN(I) = -0.5D0*RI
     GDEL = GDEL + DIRIN(I)*GS(I)
C                                     . LINEAR SEARCH ALONG -VG . . .
30 X(I) =XXS(I) + DIRIN(I)
   CALL INTOEX(X)
   CALL FCN (NPAR, G, F, U, 4)
   NFCN=NFCN+1
C                                     . QUADR INTERP USING SLOPE GDEL
   DENOM = 2.0D0*(F-AMIN-GDEL)
   IF (DENOM .LE. 0.0D0) GO TO 35
   SLAM = -GDEL/DENOM
   IF (SLAM .GT. SLAMAX) GO TO 35
   IF (SLAM .LT. SLAMIN) SLAM=SLAMIN
   GO TO 40
35 SLAM = SLAMAX
40 IF (DABS(SLAM-1.0D0) .LT. 0.1D0) GO TO 70
   DO 45 I= 1, NPAR
45 X(I) =XXS(I) + SLAM*DIRIN(I)
   CALL INTOEX(X)
   CALL FCN(NPAR,G,F2,U,4)
   NFCN = NFCN + 1
C                                     . QUADR INTERP USING 3 POINTS
   AA = FS/SLAM
   BB = F/(1.0D0-SLAM)
   CC = F2/ (SLAM*(SLAM-1.0D0))
   DENOM = 2.0D0*(AA+BB+CC)
   IF (DENOM .LE. 0.0D0) GO TO 48
   TLAM = (AA*(SLAM+1.0D0) + BB*SLAM + CC)/DENOM
   IF (TLAM .GT. TLAMAX) GO TO 48
   IF (TLAM .LT. TLAMIN) TLAM=TLAMIN
   GO TO 50
48 TLAM = TLAMAX
50 CONTINUE
   DO 51 I= 1, NPAR
51 X(I) = XXS(I)+TLAM*DIRIN(I)
   CALL INTOEX(X)
   CALL FCN(NPAR,G,F3,U,4)
   NFCN = NFCN + 1
   IF (F.GE.AMIN .AND. F2.GE.AMIN .AND. F3.GE.AMIN) GO TO 200
   IF (F .LT. F2 .AND. F .LT. F3) GO TO 61
   IF (F2 .LT. F3) GO TO 58
55 F = F3
   SLAM = TLAM

```



```

      GO TO 65
58 F = F2
      GO TO 65
61 SLAM = 1.0D0
65 DO 67 I= 1, NPAR
      DIRIN(I) = DIRIN(I)*SLAM
67 X(I) = XXS(I) + DIRIN(I)
70 AMIN = F
      ISW(2) = 2
      IF (SIGMA+FS-AMIN .LT. ROSTOP) GO TO 170
      IF (SIGMA+RHO2+FS-AMIN .GT. APSI) GO TO 75
      IF (TRACE .LT. VTEST) GO TO 170
75 CONTINUE
      IF (NFCN-NPFN .GE. NFCNMX) GO TO 190
      ITER = ITER + 1
      IF (ISWTR.GE. 3 .OR.(ISWTR.EQ. 2 .AND. MOD(ITER,10) .EQ.1))
      *      CALL MPRINT(0,AMIN)
C      . . . GET GRADIENT AND SIGMA .
      IF (ISW(3) .NE. 1) GO TO 80
      CALL INTOEX(X)
      CALL FCN(NPAR,G,AMIN,U,IFLAG)
      NFCN = NFCN + 1
80 CALL DERIVE(G,G2)
      RHO2 = SIGMA
      SIGMA = 0.0D0
      GVG = 0.0D0
      DELGAM = 0.0D0
      DO 100 I= 1, NPAR
      RI = 0.0D0
      VGI = 0.0D0
      DO 90 J= 1, NPAR
      VGI = VGI + V(I,J)*(G(J)-GS(J))
90 RI = RI + V(I,J) *G (J)
      R(I) = RI * 0.5D0
      VG(I) = VGI*0.5D0
      GAMI = G(I) - GS(I)
      GVG = GVG + GAMI*VG(I)
      DELGAM = DELGAM + DIRIN(I)*GAMI
100 SIGMA = SIGMA + G(I)*R(I)
      IF (SIGMA .LT. 0.0D0) GO TO 1
      IF (GVG .LE. 0.0D0) GO TO 105
      IF (DELGAM .LE. 0.0D0) GO TO 105
      GO TO 107
105 IF (SIGMA .LT. 0.1D0*ROSTOP) GO TO 170
      GO TO 1
107 CONTINUE
C      . UPDATE COVARIANCE MATRIX
      TRACE=0.0D0
      DO 120 I= 1, NPAR
      VII(I) = V(I,I)
      DO 120 J=1,NPAR
      D = DIRIN(I)*DIRIN(J)/DELGAM - VG(I)*VG(J)/GVG
120 V(I,J) = V(I,J) + 2.0D0*D
      IF (DELGAM .LE. GVG) GO TO 135
      DO 125 I= 1, NPAR

```

```

125 FLNU(I) = DIRIN(I)/DELGAM - VG(I)/GVG
    DO 130 I= 1, NPAR
    DO 130 J= 1, NPAR
130 V(I,J) = V(I,J) + 2.0D0*GVG*FLNU(I)*FLNU(J)
135 CONTINUE
    DO 140 I= 1, NPAR
140 TRACE = TRACE + ((V(I,I)-VII(I))/(V(I,I)+VII(I)))**2
    TRACE = DSQRT(TRACE/PARN)
    IF (ISWTR .GE. 4) CALL MATOUT(TRACE, 0)
    CALL UCOPY(X,XXS,NPAR)
    CALL UCOPY(G,GS,NPAR)
    FS = F
    GO TO 24
C
    . . . . . END MAIN LOOP
170 WRITE(ISYSWR,500)
    ISW(2) = 3
    IF(ISWTR .GE. 0) CALL MPRINT(1-ITAU,AMIN)
    ISWTR = ISWTR - 3*ITAU
    IF (ISWTR .GT. 0) CALL MATOUT(TRACE,1)
    IF (ITAU .GT. 0) GO TO 435
    IF (MATGD .GT. 0) GO TO 435
    NPARGD = NPAR*(NPAR+5)/2
    IF (NFCN-NPFN .GE. NPARGD) GO TO 435
    WRITE (ISYSWR,180)
180 FORMAT ( 55H COVARIANCE MATRIX INACCURATE. MINUIT WILL RECALCULA
1TE)
    CALL HESSE
    CALL MPRINT(1,AMIN)
    CALL MATOUT(0.0D0, 1)
    IF (ISW(2) .GE. 2) ISW(2) = 3
    GO TO 435
190 ISW(1) = 1
    GO TO 230
200 WRITE (ISYSWR,650)
    CALL UCOPY(XXS,X,NPAR)
    ISW(2) = 1
    IF (SIGMA .LT. ROSTOP) GO TO 170
    IF (MATGD .GT. 0) GO TO 2
230 WRITE (ISYSWR,510)
    CALL INTOEX(X)
    CALL MPRINT(1-ITAU, AMIN)
    ISWTR = ISW(5) - ITAU*3
    IF (ISWTR .LT. 1) GO TO 435
    IF (ISW(2) .LE. 1) GO TO 435
    CALL MATOUT(TRACE,1)
435 RETURN
470 FORMAT ( 37H START MIGRAD MINIMIZATION. 8X 66HCONVERG
1ENCE CRITERIA — ESTIMATED DISTANCE TO MINIMUM (EDM) .LT.E9.2/45
2X 11HOR EDM .LT.E9.2, 46H AND FRACTIONAL CHANGE IN VARIANCE MATR
3IX .LT.E9.2)
500 FORMAT ( 34H MIGRAD MINIMIZATION HAS CONVERGED )
510 FORMAT ( 39H MIGRAD TERMINATED WITHOUT CONVERGENCE )
520 FORMAT (11X 43HCOVARIANCE MATRIX IS NOT POSITIVE-DEFINITE.)
650 FORMAT ( 34H MIGRAD FAILS TO FIND IMPROVEMENT )
    END

```

```

C ***** MINNEW *****
      SUBROUTINE MINNEW
      IMPLICIT REAL*8 (A-H,O-Z)
CC      THIS IS THE MAIN PROGRAM, DISGUISED AS A SUBROUTINE FOR
CC      REASONS OF COMPATIBILITY BETWEEN SYSTEMS.  IT INITIALIZES
CC      SOME CONSTANTS IN COMMON (INCLUDING THE LOGICAL I/O UNIT NOS.)
CC      WHICH WOULD HAVE TO BE IN BLOCK DATA FOR MANY COMPILERS.
CC      THEN VERIFIES THAT FCN GIVES THE SAME VALUE WHEN CALLED
CC      TWICE WITH THE SAME ARGUMENTS, AND PASSES CONTROL TO COMAND.
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /DERIVA/ G(30) ,G2(30)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAUT ,SIGMA
      INTEGER CWORD,CWORD2,CWORD3
      INTEGER RET1,RET2,RET3
      DATA RET1,RET2,RET3/ 4H RET, 4H RETU, 4H RETR /
210  FORMAT ( 5( 8X,2A4,A2))
110  CONTINUE
      NFCN = 1
      CALL MIDATA
      CALL INTOEX(X)
C      WRITE (ISYSWR,120)
120  FORMAT (/20H0FIRST ENTRY TO FCN )
C      CALL FCN(NPAR,G,AMIN,U,1)
      CALL FCN(NPAR,G,AMIN,U,4)
      CALL MPRINT(1,AMIN)
      CALL FCN(NPAR,G,F ,U,4)
      IF (F .NE. AMIN) GO TO 160
      NFCN = 3
      CALL COMAND
      IF (CWORD2.EQ.RET1 .OR.CWORD2.EQ.RET2 .OR.CWORD2.EQ.RET3) RETURN
      GO TO 110
160  CONTINUE
      WRITE (ISYSWR,880) AMIN, F
      STOP
880  FORMAT ( 62H0FOR THE ABOVE VALUES OF THE PARAMETERS, FCN IS TIME-
1DEPENDENT/ 4H0F = E22.14, 15H FOR FIRST CALL/ 4H F =E22.14,
211H FOR SECOND)
      END
C ***** MINOS *****
      SUBROUTINE MINOS
      IMPLICIT REAL*8 (A-H,O-Z)
CC      PERFORMS A MINOS ERROR ANALYSIS ON THOSE PARAMETERS FOR
CC      WHICH IT IS REQUESTED ON THE MINOS COMMAND CARD.
CC      THE PARAMETER IN QUESTION IS VARIED, AND THE MINIMUM OF THE
CC      FUNCTION WITH RESPECT TO THE OTHER PARAMETERS IS FOLLOWED
CC      UNTIL IT CROSSES THE VALUE FMIN+UP.
CC
      REAL DOIBLANK '/'
      COMMON /MINERR/ ERP(30) ,ERN(30)
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR

```

```

INCLUDE 'PAREXT.CCC'
COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
COMMON /VARIAN/ V(15,15)
COMMON /UNIT / ISYSRD      ,ISYSWR      ,ISYSPU
COMMON /TITLE / TITLE(13) ,DATE(2)      ,ISW(7)      ,NBLOCK
COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ      ,NFCN      ,NFCNMX
COMMON /CARD / IDUM,CWORD      ,CWORD2      ,CWORD3      ,WORD7(7)
COMMON /MINIMA/ AMIN      ,UP      ,NEWMIN      ,ITAU      ,SIGMA
COMMON /VARIAT/ VT(15,15)
COMMON /DERIVA/ G(30)      ,G2(30)
DIMENSION W(30),SPT(3,20)
DIMENSION LMI(30), KIRSCH(5),XDEV(15)
INTEGER CWORD,CWORD2,CWORD3
DATA APOS,ANEG,ADOT,ASTAR / 4HPOSI,4HNEGA,1H., 1H* /
IF (NPAR .LE. 0) GO TO 700

```

C

. . UNPACK PARAMETER REQUESTS .

```

MARC = 0
DO 5 I=1,30
5 LMI(I)=0
KNT = 0
DO 20 I= 2, 7
LIME = WORD7(I) + 0.5D0
IF (LIME .EQ. 0) GO TO 20
DO 10 K= 1, 5
K2 = 6 - K
LEMON = MOD(LIME,100)
KIRSCH(K2) = LEMON
10 LIME = LIME/100
DO 15 J= 1, 5
LEMON = KIRSCH(J)
IF (LEMON .GT. MAXEXT) GO TO 14
IF (LEMON .EQ. 0) GO TO 15
IF (LCORSP(LEMON) .EQ. 0) GO TO 14
KNT = KNT + 1
LMI(KNT) = LEMON
GO TO 15
14 MARC = 1
15 CONTINUE
20 CONTINUE
IF (KNT .GT. 0) GO TO 40
DO 30 I= 1, MAXEXT
IF (LCORSP(I) .LT. 1) GO TO 30
KNT = KNT + 1
LMI(KNT) = I
IF (KNT .EQ. 30) GO TO 40
30 CONTINUE
40 CONTINUE
IF (MARC .EQ. 1) WRITE (ISYSWR,811)
WRITE (ISYSWR, 810) (LMI(IZ3),IZ3=1,KNT)

```

C

. . SAVE AND PREPARE START VALS

```

SIGSAV = SIGMA
TOLER = EPSI
APSI = EPSI*0.5D0
ITAU = 1
ABEST=AMIN

```

```

    AIM = AMIN + UP
    NSAVE = NFCNMX
    MPAR=MPAR
    DO 130 I= 1, MPAR
      XT(I) = X(I)
    DO 125 J= 1, MPAR
125  VT(I,J) = V(I,J)
130  CONTINUE
      DO 135 I= 1, NU
        ERP(I) = 0.000
        ERN(I) = 0.000
135  W(I) = WERR(I)
      KNT = 0
C
      . . . . . START MAIN LOOP . . . . .
150  KNT = KNT + 1
      ISW(1) = 0
      NLIMIT = NFCN + NSAVE
      IF (KNT .GT. 30) GO TO 590
      IF (LMI(KNT) .LT. 1) GO TO 590
      ILAX = LMI(KNT)
      ERP(ILAX) = 0.000
      IT = LCORSP(ILAX)
      XTIT = XT(IT)
      CALL INTOEX(XT)
      UT = U(ILAX)
      SPT(1,1) = UT
      SPT(2,1) = AIM
      SPT(3,1) = doIBLANK
      SPT(1,2) = UT
      SPT(2,2) = ABEST
      SPT(3,2) = ASTAR
      SPT(1,3) = UT
      SPT(2,3) = AIM + 0.200*(AIM-ABEST)
      SPT(3,3) = ADOT
      NSPT = 3
      IF (LCODE(ILAX) GT. 1) GO TO 160
      ALIM(ILAX) = UT -100.00*W(ILAX)
      BLIM(ILAX) = UT +100.00*W(ILAX)
160  CONTINUE
      XUNIT = DSQRT(UP/VT(IT,IT))
      MARC = 0
      DO 162 I= 1, MPAR
        IF (I .EQ. IT) GO TO 162
        MARC = MARC + 1
        XDEV(MARC) = XUNIT*VT(IT,I)
162  CONTINUE
      CALL FIXPAR (IT,1,ILAX)
      SIG = 1.000
      ASIG = APOS
      DULIM = BLIM(ILAX) - UT
      IF(ISW(2).LT.1) GO TO 460
C
      . . . . . SIG=SIGN OF ERROR BEING CALCD
165  WRITE (ISYSWR,206)  ASIG,ILAX,NAM(ILAX)
      ITER = 0
      LIMSET = 0

```

```

DU1 = SIG*W(ILAX)
IF (DABS(DU1) .LE. DULIM) GO TO 180
LIMSET = 1
DU1 = SIG * DULIM
IF (DULIM .LT. 1.0D-3*W(ILAX)) GO TO 440
180 U(ILAX) = UT + DU1
IF (NPAR .EQ. 0) GO TO 205
FAC = DU1/W(ILAX)
DO 185 I= 1, NPAR
185 X(I) = XT(I) + FAC*XDEV(I)
200 CALL INTOEX (X)
205 WRITE (ISYSWR,801) ILAX,UT,DU1,U(ILAX)
CALL FCN(NPAR,G,AMIN,U,4)
NFCN = NFCN + 1
NFCNMX = NLIMIT - NFCN
CALL MIGRAD
IF (AMIN .LT. ABEST) GO TO 650
IF (ISW(1) .GE. 1) GO TO 450
IF (ISW(2) .GE. 2) GO TO 240
NFCNMX = NLIMIT - NFCN
CALL SIMPLX
IF (AMIN .LT. ABEST) GO TO 650
IF (ISW(1) .GE. 1) GO TO 450
NFCNMX = NLIMIT - NFCN
CALL MIGRAD
IF (AMIN .LT. ABEST) GO TO 650
IF (ISW(1) .GE. 1) GO TO 450
IF (ISW(2) .LT. 2) GO TO 460
240 CREM = AMIN - ABEST
NSPT = NSPT + 1
SPT(1,NSPT) = U(ILAX)
SPT(2,NSPT) = AMIN
SPT(3,NSPT) = ASIG
IF (CREM .LE. 0.0D0) GO TO 650
SQUC = DSQRT(UP/CREM)
IF (DABS(AMIN-AIM) .LT. TOLER) GO TO 400
C . . . ANOTHER ITERATION NECESSARY
ITER = ITER + 1
IF (ITER .GT. 6) GO TO 430
IF (ITER .EQ. 1) GO TO 270
C CHECK PREVIOUS ITERATION TO AVOID OSCILLATING
SQUC2 = SQUC
IF ((SQUC2-1.0D0) * (SQUC1-1.0D0) .GT. 0.0) GO TO 270
SQUC = 0.65D0*SQUC2 + 0.35D0
SQUC11 = 1.0D0/SQUC1
IF ((SQUC11-SQUC)*(SQUC-1.0D0) .GT. 0.0D0) GO TO 270
WRITE (ISYSWR,260)
260 FORMAT ( 44H MINOS IS HAVING TROUBLE WITH THIS PARAMETER)
SQUC = 0.5D0*SQUC11 + 0.5D0
270 CONTINUE
SQUC1 = SQUC
DU1 = DU1 * SQUC
IF (DABS(DU1) .LE. DULIM) GO TO 280
IF (LIMSET .EQ. 1) GO TO 440
LIMSET = 1

```

```

      DU1 = SIG*DULIM
230 U(ILAX) = UT + DU1
      DO 290 I= 1, NPAR
290 X(I) = XT(I) + SQUC*(X(I)-XT(I))
      GO TO 200

```

C . ERROR SUCCESSFULLY CALCULATED

```

400 EROS = DU1 * SQUC
      WRITE (ISYSWR,808) ASIG,ILAX,NAM(ILAX),EROS
410 WRITE (ISYSWR,812)
      IF (SIG .GT. 0.0D0) GO TO 420
      ERN(ILAX) = EROS
      GO TO 500
420 ERP(ILAX) = EROS
      SIG = -1.0D0
      ASIG = ANEG
      DULIM = UT - ALIM(ILAX)
      GO TO 165

```

C FAILURE RETURNS

```

430 WRITE (ISYSWR, 809)
      EROS = 0.0D0
      GO TO 410
440 WRITE (ISYSWR, 807) ASIG,ILAX,NAM(ILAX),DULIM
      EROS = 0.0D0
      GO TO 410
450 WRITE (ISYSWR, 802) NSAVE
      GO TO 500
460 WRITE (ISYSWR, 805)

```

C PARAMETER FINISHED. RESET V

```

500 CONTINUE
      CALL RESTOR(1)
      ISW(2) = 3
      DO 560 I= 1, MPAR
      DO 550 J= 1, MPAR
550 V(I,J) = VT(I,J)
560 CONTINUE
      IF (NSPT .LT. 9) GO TO 150
      NSPT3 = NSPT - 3
      WRITE (ISYSWR,813) ILAX,NAM(ILAX),NSPT3
      CALL PLTCON(NSPT,SPT)
      WRITE(ISYSWR,814) ILAX
      GO TO 150

```

C PRINTOUT FINAL VALUES .

```

590 CALL UCOPY(XT,X,MPAR)
      CALL INTOEX (XT)
      SIGMA = SIGSAV
      AMIN = ABEST
      CALL MPRINT(2,AMIN)
      CALL MATOUT(0.0D0, 1)
      GO TO 700

```

C NEW MINIMUM FOUND

```

650 NEWMIN = 1
      ISW(2) = 0
      CALL RESTOR(1)
      CALL EXTOUT(X)
      DO 670 I= 1, NPAR

```

```

670 DIRIN(I) = DSQRT(VT(I,I)*UP)
700 RETURN
801 FORMAT ( 10H0PARAMETER I4, 7H SET TOE11.3,3H + E10.3,3H = E12.3)
802 FORMAT ( 87H0CALL LIMIT EXCEEDED. CALCULATION OF MINOS ERROR FOR
1 THIS PARAMETER REQUIRES MORE THAN I5, 11H FCN CALLS./)
805 FORMAT (/45X 46HMINOS ERROR NOT CALCULATED FOR THIS PARAMETER.)
806 FORMAT ( 18H0DETERMINATION OF A4, 30HTIVE MINOS ERROR FOR PARAME
1TER I3, 2X A10)
807 FORMAT (/45X4HTHE A4, 29HTIVE MINOS ERROR OF PARAMETER I3, 2H, A
110, 9H, EXCEEDS E12.4)
808 FORMAT (/45X4HTHE A4, 29HTIVE MINOS ERROR OF PARAMETER I3, 2H, A
110, 4H, IS E12.4)
809 FORMAT (/45X 48HTOO MANY ITERATIONS. SEE PLOT BELOW FOR DETAILS.)
810 FORMAT ( 38H MINOS ERRORS REQUESTED FOR PARAMETERS30I3//)
811 FORMAT ( 55H THERE ARE MISTAKES IN THE MINOS COMMAND CARD JUST RE
1AD)
812 FORMAT (45X, 75(1H*))
813 FORMAT (50H1MINOS FINDS NON-PARABOLIC BEHAVIOR FOR PARAMETER ,I3,
12H, A10/28H ERROR CALCULATION REQUIRED ,I3.20H MINIMIZATIONS.
243HA SUMMARY OF POINTS FOUND IS PLOTTED BELOW. /
313H0 FCN VALUE )
814 FORMAT (20H+ PARAMETER I3,1H,/)
END

```

```

C ***** MINUTL *****
SUBROUTINE MINUTL

```

```

CENDIF

```

```

CALL MINNEW
RETURN
END

```

```

C ***** MLIST *****
SUBROUTINE MLIST (M,J,K,MI)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

C READS K ELEMENT NAMES FROM THE JTH BCD LOCATION OF M AND STORES
C THEIR INDICES IN MI.

```

```

INTEGER ELNUM
DIMENSION MI(1)
CALL DATA(M,2,J,K,MI)
DO 1 I=1,K
1 MI(I)=ELNUM(MI(I))

```

```

RETURN
END

```

```

C***** MMM *****
SUBROUTINE MMM(M,KA,MI)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

C MATRIX MULTIPLICATION. R(M)=R(MK)*...R(M2)*R(M1)
C IF MI(I) IS NEGATIVE, R(M)=R(MK)...(1/R(MI))..R(M2).R(M1) ETC.

```

```

C SYNCH INSTRUCTION MMM>

```

```

C X MMM A B C . . . . .
C A,B,C,...ARE PREVIOUSLY DEFINED MATRICES, OR BML;S, OR NUMBERED
C PARENTHESES. THIS INPUT LIST IS EXPANDED BY MIFILL AND MIEXP TO
C FORM A LIST OF MATRIX INDICES, STORED IN MI(I). THESE ARE
C MULTIPLIED TOGETHER TO FORM MATRIX NAMED X.

```

```

INCLUDE 'BSTORE.CCC'

```



```

INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'DIM.CCC'
INTEGER doIR7/'R7'  '/,doIROT/'ROT' '/,doIUNO/'(1)' '/'
INTEGER doIMAP /'MAP' '/'
INTEGER DOLMXY /'MXY'/'
DIMENSION MI(1)
INTEGER ELNUM

```

C NT IS WORKING STORAGE FOR MATRIX

```

KN = INFF(20,M)
KK = 3
IF (KN.EQ.doIR7.OR.KN.EQ.doIROT.OR.KN.EQ.doIMAP) KK = 7

```

3 NT = M7END

K = 3

NT = MEND

DO 100 I=1,KA

IF (INFF(20,JIABS(MI(I))).EQ.DOLMXY) GO TO 100

K = 7

NT = M7END

GO TO 5

100 CONTINUE

5 M1= MI(1)

C IF FIRST INDEX IN M1LIST IS -, SET IT + AND INVERT

IF(M1)1,2,2

1 M1=-M1

CALL INV(NT,M1)

K1=2

GO TO 4

2 CONTINUE

IF (KA.GE.2) GO TO 7

C ONLY ONE ELEMENT. MULTIPLY BY UNIT MATRIX NAMED (1)

MUNIT = doIUNO

M2 = ELNUM(MUNIT)

GO TO 8

7 M2 = MI(2)

IF (KA.GT.2) GO TO 12

8 CONTINUE

C IF THERE ARE ONLY 2 MATRIX MULT.,RESULT SHOULD BE STORED FOR M.

CALL MXD(M,M1,M2,KK)

GO TO 11

C ACCUM. RESULTS OF MULT. ARE STORED IN WORKING STORAGE OF NT

12 CALL MXD(NT,M1,M2,K)

K1=3

4 DO 10 I=K1,KA

M1=MI(I)

IF (I.EQ.KA) GO TO 6

CALL MXD(NT,NT,M1,K)

GO TO 10

C ON LAST ITERATION, FINAL MATRIX STORED FOR M.

6 CALL MXD(M,NT,M1,KK)

10 CONTINUE

11 CONTINUE

RETURN

END

C ***** MMRSV *****

SUBROUTINE MMRSV(M0,NAME0,KA0,KB0,NAME1)

IMPLICIT REAL*8 (A-H,O-Z)

INCLUDE 'SWTCH.CCC'

INTEGER R7

DATA MXPMY,R7/1,5/

INTEGER NEGZER/-32767/,doIMMM/'MMM '/

DATA IBLNK/4H /

INM = doIMMM

CALL STINFO(M0,NAME0,INM,KA0,KB0,NAME1)

KN = IBLNK

IF (MSIZE.EQ.7) KN=R7

IF (MSIZE.EQ.3) KN= MXPMY

C SET KIND IF INDICATED BY MSIZE

IF (NAME1 .NE. 0) GO TO 1

IF (KN .EQ. IBLNK) KN = MXPMY

CALL INP6(M0,KN,NEGZER,1,-2,0)

RETURN

1 CALL INP6(M0,KN,NEGZER,KA0,2,0)

RETURN

END

C ***** MNAME *****

FUNCTION MNAME(M)

IMPLICIT REAL*8 (A-H,O-Z)

C FINDS NAME OF ELEMENT WHOSE NUMBER IS M.

C IF M IS NEGATIVE,RETURN/NAME FOR INVERSE

INCLUDE 'BINFF.CCC'

INCLUDE 'FLTN.CCC'

MNAME = INFF(2,IABS(M))

IF (M.GT.0) RETURN

MN=SHFTR(MNAME,8)

MNAME = LOR(IFL(15),MN)

RETURN

END

C *****MODQ*****

SUBROUTINE MODQ (M,N,V,W)

IMPLICIT REAL*8 (A-H,O-Z)

C

INCLUDE 'DIM.CCC'

INCLUDE 'CONST.CCC'

COMMON/MJ/ML(3)

DIMENSION RX(2,3),RY(2,3),R(2,3),RYW(3),RPQ(4),RW(3)

DIMENSION V(1),W(7),PAR(6),T(7,7),D(7,7)

EQUIVALENCE (BR,PAR(3))

DATA RADEG/.0174532925/

C

CALL DATA (M,1,1,6,PAR)

BR = PAR(3)*(1.D0+V(6))

CALL MAGNET (N,PAR)

CALL RTRV7(N,T,RW)

CALL MXV7(T,V,W)

S = PAR(1)

RHOIX = (V(2)-W(2))/S

```

RHOIY = (V(4)-W(4))/S
ML(1) = M7END - 8
ML(2) = N
ML(3) = M7END - 9
CALL RTRV7 (MUNIT,D,RW)
D(2,1) = -V(4)*RHOIY
D(4,3) = -V(2)*RHOIX
CALL STOR7(M7END-8,D,RW)
D(2,1) = W(4)*RHOIY
D(4,3) = W(2)*RHOIX
CALL STOR7(M7END-9,D,RW)
CALL RXY(N,RX,RY,RW)
IF (RHOIX.EQ.0.) GO TO 3

```

C

```

SGN = DSIGN(1.D0,RHOIX)/RADEG
PAR(1) = -PAR(1)
PAR(4) = BR*RHOIX
PAR(5) = SGN*V(2)
PAR(6) = -SGN*W(2)
CALL MAGNET(N,PAR)
CALL RXY(N,RX,R,RW)

```

```

3 IF (RHOIY.EQ.0.) GO TO 4

```

C

```

SGN = DSIGN(1.D0,RHOIY)/RADEG
PAR(1) = -PAR(1)
PAR(2) = -PAR(2)
PAR(4) = BR*RHOIY
PAR(5) = SGN*V(4)
PAR(6) = -SGN*W(4)
CALL MAGNET(N,PAR)
CALL RXY(N,RY,R,RYW)
RW(3) = RW(3) + RYW(3)

```

C

```

4 CALL CNVMAT(7,T,RX,RY,RW,RPQ)
CALL STOR7(N,T,RW)
CALL MMM(N,3,ML)
RETURN
END

```

C*****MOVE*****

```

SUBROUTINE MOVE(M,N,V)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

INCLUDE 'BINFF.CCC'
INCLUDE 'DIM.CCC'
INCLUDE 'MOVTOG.CCC'
common/seed/nr

```

```

DIMENSION T(7,7),RW(3),RW1(3),PAR(10),ML(7),V(7)
COMMON/MK/ML
DIMENSION PARR(6)
DIMENSION PP(6)
REAL*8 L,K
INTEGER BDAT

```

```

EQUIVALENCE (PAR(1),HX),(PAR(2),DHX),(PAR(3),HY),(PAR(4),DHY),
1 (PAR(5),HZ),(PAR(6),THETZ)

```

```

2      ,(PAR(7),XIN),(PAR(8),YIN),(PAR(9),XOUT),(PAR(10),YOUT)
  DIMENSION W(7),PAT(4)
  LOGICAL QDSW,SXSW
  DATA RW1/3*0.0/
  DATA IBLNK/1H /

```

```

C  MISALIGNMENT OF ELEMENTS —SIX DEGREES OF FREEDOM.
C  USES 7X7 FORMALISM IN CONJUNCTION WITH FXPT, TRK AND CYEM.
C  CLOSED ORBIT FOUND BY FXPT RATHER THAN CYX WHICH NEEDS MAGS.  EXAMPLE

```

```

C .....
C  QF  MAG      //L      /G      /BRHO
C  H   VEC 6    //HX      /DHX      /HY      /DHY      /HZ      /THETZ

```

```

C  —FORMAT 1—

```

```

C  QF*  MOVE      //QF      /H

```

```

C .....

```

```

C  —FORMAT 2.0—

```

```

C  QF*  MOVE      //QF      /HX      /HY      /HZ      /THETZ

```

```

C  —FORMAT 2.1—

```

```

C  QF*  MOVE      //QF  V /HX      /DHX      /HY      /DHY

```

```

C .....

```

```

C  M = INDEX OF MOVE COMMAND.  N=INDEX OF ELEMENT AT WHICH MATRIX

```

```

C                                FOR M TO BE STORED.

```

```

C  V = ORBIT MAPPED.  MQ=INDEX OF ELEMENT MOVED.  MPAR=INDEX OF VEC.

```

```

C  KB.GE.1 — PARAMETERS RANDOM.  KB.GE.2—RANDOM SEED RESET SO NEXT

```

```

C  HX,HY,HZ ARE TRANSLATIONS ALONG X,Y,Z AXES,

```

```

C  DHX, DHY ARE ROTATIONS ABOUT THE Y,X AXES, I.E. TILTS IN X-Z, Y-Z

```

```

C  THETZ IS ROTATION ABOUT Z AXIS.  Z = BEAM DIR., X = HOR., Y = VERT.

```

```

C  ORDER OF MISALIGNMENTS—LONGITUDINAL(Z), TRANSVERSE(X,Y), Z-ROTATION

```

```

C  M IS THE INDEX OF THE MOVE COMMAND.

```

```

C  N IS THE INDEX OF THE ELEMENT AT WHICH THE MATRIX CORRESPONDING

```

```

C  TO M IS TO BE STORED.

```

```

C  V IS THE ORBIT TO BE MAPPED BY THE MATRIX T CORRESPONDING TO M.

```

```

C  MQ IS THE INDEX OF THE ELEMENT TO BE MOVED.

```

```

C  MPAR IS THE INDEX OF THE VEC COMMAND THAT CONTAINS THE MISALIGNMENTS.

```

```

C  KB GE.1 — PARAMETERS RANDOM

```

```

C  KB GE.2 — RANDOM SEED RESET SO NEXT PASS IS SAME

```

```

  MQ=MDAT(M,1)

```

```

  KB=INFF(5,M)

```

```

  IF (KB.LT.2) GO TO 10

```

```

  IF (TOG) nrrn=nr

```

```

10  QDSW = .FALSE.

```

```

  KIND = INFF(20,M)

```

```

  IF (KIND.EQ.3HMAP) GO TO 3

```

```

  NOPQ=INFF(1,MQ)

```

```

  SXSW = .FALSE.

```

```

  IF (NOPQ.NE.3HMAG) GO TO 12

```

```

  CALL DATA (MQ,1,1,6,PAR)

```

```

  IF (PAR(4).EQ.0.) QDSW = .TRUE.

```

```

  L=PAR(1)

```

```

  K=PAR(2)/PAR(3)/(1.D0+V(6) )

```

```

  GO TO 11

```

```

3  SXSW = .TRUE.

```

```

  CALL DATA (MQ,1,1,4,PAT)

```

```

      L = PAT(1)
      GO TO 11
12    CALL RTRV7 (MQ,T,RW)
      L=RW(1)
      K = 0.
11    IF (M.NE.N) GO TO 7
      NAVEC = BDAT(M,2)
      IF (NAVEC.EQ.IBLNK) GO TO 7
      MVEC = MDAT(M,2)
      CALL DATA (MVEC,5,1,7,V)
7     NAPAR = BDAT(M,3)
      MPAR = M
      HZ = 0.
      THETZ = 0.
      NVAR5 = 4
      IF (NAPAR.EQ.IBLNK) GO TO 1
      NVAR5 = 6
      MPAR = MDAT(M,3)
1     CALL DATA(MPAR,1,1,NVAR5,PARR)
      DO111 I=1,NVAR5
      PAR(I)=PARR(I)
      IF (KB.GE.1) PAR(I)=PAR(I)*(RAN(nr)-.5)
111   CONTINUE
      TLX=L*DHX/2.D0
      HX1=HX-TLX
      HX2=HX+TLX
      TLY=L*DHY/2.D0
      HY1=HY-TLY
      HY2=HY+TLY

2     ML(1)=M7END-1
      ML(2)=M7END-2
      ML(3)=M7END-6

      CALL DRIFT(M7END-1,HZ)

      CALL RTRV7(MUNIT,T,RW)
      T(1,7)=-HX1
      T(2,7)=-DHX
      T(3,7)=-HY1
      T(4,7)=-DHY
      CALL STOR7(M7END-2,T,RW1)

      IF (HZ.EQ.0..AND.THETZ.EQ.0.) GO TO 201
      CALL DRIFT(M7END-1,HZ)
      CALL ROTZ(THETZ,T)
      CALL STOR7(M7END-6,T,RW1)

      CALL MMM(M7END-2,3,ML)

201   CONTINUE
      CALL RTRV7(M7END-2,T,RW)
      CALL MXV7(T,V,V)
      IF (.NOT.QDSW) GO TO 20
      CALL MODQ (MQ,N,V,W)

```

```

      GO TO 8
20  IF (.NOT.SXSW) GO TO 21
      CALL SXTP(N,V,PAT)
      GO TO 14
21  CALL DATA(MQ,1,1,6,PP)
      PP(3) = PP(3)*(1.D0+V(6))
      PP(4) = PP(4)+PP(2)+(V(1)+V(2)*L/2.D0)
      CALL MAGNET(N,PP)
      CALL RTRV7(N,T,RW)
      CALL MXV7(T,V,W)
C
8   XIN=V(1)
      YIN=V(3)
      XOUT=W(1)
      YOUT=W(3)
      DO 112 I=1,NVARS
112  PAR(I)=PARR(I)
      CALL STDAT(M,1,1,10,PAR)
      DO 9 I=1,7
9    V(I) = W(I)

14  CALL RTRV7 (MUNIT,T,RW)
      T(1,7)=-HX2
      T(2,7)=-DHX
      T(3,7)=-HY2
      T(4,7)=-DHY
      CALL STOR7(M7END-7,T,RW)

      ML(1)=- (M7END-6)
      ML(2) = - (M7END-7)
      ML(3)=- (M7END-1)
      IF (HZ.EQ.0..AND.THETZ.EQ.0.) ML(1) = MUNIT
      IF (ML(1).EQ.MUNIT) ML(3) = MUNIT
      CALL MMM(M7END-7,3,ML)

      ML(1)=M7END-2
      ML(2)=N
      ML(3) = M7END - 7

      CALL RTRV7(M7END-7,T,RW)
      CALL MXV7(T,V,V)
      IF(N.EQ.0) GO TO 113
      CALL MMM(N,3,ML)
      IF(M.NE.N) GO TO 113
      CALL RTRV7(N,T,RW)
      CALL ROW5(T)
      CALL STOR7(N,T,RW)

113 IF (KB.LT.2) RETURN
      IF (TOG) GO TO 115
      TOG = .TRUE.
      RETURN
115 nr=nrn
      TOG = .FALSE.
      RETURN

```

```

      END
C ***** MPRINT *****
      SUBROUTINE MPRINT (IKODE,FVAL)
      IMPLICIT REAL*8 (A-H,O-Z)
CC      PRINTS THE VALUES OF THE PARAMETERS AT THE TIME OF THE CALL.
CC      ALSO PRINTS OTHER RELEVANT INFORMATION SUCH AS FUNCTION VALUE,
CC      ESTIMATED DISTANCE TO MINIMUM, PARAMETER ERRORS, STEP SIZES.
CC      ACCORDING TO THE VALUE OF IKODE,THE PRINTOUT IS LONG FORMAT,
CC      SHORT FORMAT, OR MINOS FORMAT (0,1,2)
CC
      COMMON /MINERR/ ERP(30) ,ERN(30)
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
C      . GET TIME AND PRINT HEADINGS .
      TI = FTIME(BIDON)
      IF (IKODE .EQ. 2) GO TO 10
      WRITE (ISYSWR,1000)
      GO TO 18
C      IKODE = 2 MINOS PRINTOUT HAS SPECIAL HEADINGS.
10 WRITE (ISYSWR,1006) TITLE,DATE
18 E = SIGMA
      KOUNT = 0
C      . . . LOOP OVER PARAMETERS . .
      DO 200 I= 1, NU
      IF (NAM (I)) 20,200,20
20 L = LCORSP(I)
      IF (L .EQ. 0) GO TO 55
C      VARIABLE PARAMETER. CALCULATE EXTERNAL ERROR IF V EXISTS
      IF (ISW(2) .LT. 1) GO TO 27
      DX = DSQRT(DABS(V(L,L)*UP))
      IF (LCODE(I) .LE. 1) GO TO 26
      AL = ALIM(I)
      BA = BLIM(I) - AL
      DU1 = AL + 0.5D0 *(DSIN(X(L)+DX) +1.0D0) * BA - U(I)
      DU2 = AL + 0.5D0 *(DSIN(X(L)-DX) +1.0D0) * BA - U(I)
      IF (DX .GT. 1.0D0) DU1 = BA
      DX = 0.5D0 * (DABS(DU1) + DABS(DU2))
26 WERR(I) = DX
27 X1 = X(L)
      X2 = DIRIN(L)
      IF (IKODE .LT. 2) GO TO 29
      X1 = ERP(I)
      X2 = ERN(I)
29 IF (KOUNT) 30,30,40
30 KOUNT = 1
      WRITE (ISYSWR,1001) FVAL,NFCN,TI,E, L,I,NAM(I),NMIND(I),U(I),
1 WERR(I) ,X1,X2
      GO TO 45
40 WRITE (ISYSWR,1002) L,I,NAM(I),NMIND(I),U(I),WERR(I),X1,X2

```

```

45 IF (LCODE(I) .LE. 1) GO TO 200
   IF (DABS(DCOS(X(L))) .LT. 0.001D0) WRITE (ISYSWR,1004)
   GO TO 200
C           FIXED PARAMETER. PRINT ONLY IF IKODE .GT.0
55 IF (IKODE .EQ. 0) GO TO 200
   IF (KOUNT) 60,60,70
60 KOUNT = 1
   WRITE (ISYSWR,1001) FVAL,NFCN,TI,E, L,I,NAM(I),NMIND(I),U(I)
   GO TO 200
70 WRITE (ISYSWR,1003) I,NAM(I),NMIND(I),U(I)
200 CONTINUE
   IF (IKODE.GE.1 .AND.ISW(2).GE.1) WRITE (ISYSWR,1005) UP
   RETURN
1000 FORMAT(/ 4X9HFCN VALUE5X5HCALLS4X4HTIME4X6H EDM 4X 75HINT.EXT.
1 PARAMETER VALUE ERROR INTERN.VALUE INT.STEP SIZ
2E)
1001 FORMAT ( E15.7,I7,F9.3, E11.2, I6,I4,1XA5,A5,4E14.5)
1002 FORMAT (1H 41XI6 I4,1XA5,A5,4E14.5)
1003 FORMAT (1H 47X I4,1XA5,A5,4E14.5)
1004 FORMAT (1H 52X 42HWARNING - - ABOVE PARAMETER IS AT LIMIT.)
1005 FORMAT (/45X 40HERRORS CORRESPOND TO FUNCTION CHANGE OF F10.4)
1006 FORMAT (1H140X36HRESULTS OF FULL MINOS ERROR ANALYSIS/41X36(1H*)/
1 /20X 15A4//80X 9HPARABOLIC4X 26H..... MINOS ERRORS ....
2.. / 4X9HFCN VALUE5X5HCALLS4X4HTIME4X6H EDM 4X 75HIN
3T.EXT. PARAMETER VALUE ERROR POSITIVE NEGA
4TIVE )
END
C ***** MPUNCH *****
SUBROUTINE MPUNCH
IMPLICIT REAL*8 (A-H,O-Z)
CC PUNCHES CURRENT PARAMETER VALUES AND STEP SIZES ONTO CARDS
CC IN FORMAT WHICH CAN BE REREAD BY MINUIT FOR RESTARTING.
CC THE COVARIANCE MATRIX IS ALSO PUNCHED IF IT EXISTS.
CC
COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
INCLUDE 'PAREXT.CCC'
COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
COMMON /VARIAN/ V(15,15)
COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
DIMENSION VC(7)
TI = FTIME(DUMMY)
WRITE (ISYSPU,1002)
WRITE (ISYSPU,1000) TITLE,DATE,TI
DO 200 I= 1, NU
  IF (NAM (I)) 20,200,20
C (AVOID PUNCHING MINUS ZERO)
20 IF (WERR(I) .EQ. 0.0) WERR(I) = 0.0
  IF (LCODE(I) .GT. 1) GO TO 100
C PARAMETER WITHOUT LIMITS
  WRITE (ISYSPU,1001) I,NAM(I),U(I),WERR(I)
  GO TO 200
C PARAMETER WITH LIMITS
100 CONTINUE
  WRITE (ISYSPU,1001) I,NAM(I),U(I),WERR(I),ALIM(I),BLIM(I)

```



```

200 CONTINUE
  WRITE (ISYSPU,1002)
  IF (ISW(2) .LT. 3) GO TO 300
  WRITE (ISYSPU,1003) NPAR
  K = 0
  KC = 0
  DO 250 I= 1, NPAR
  DO 250 J= 1, NPAR
    K = K + 1
    VC(K) = V(I,J)
    IF (K .NE. 7) GO TO 250
    K = 0
    KC = KC + 1
    WRITE(ISYSPU,1004) VC,TI,NBLOCK,KC
250 CONTINUE
  IF (K .EQ. 0) GO TO 300
  KP1 = K + 1
  DO 260 I= KP1, 7
260 VC(I) = 0.
  KC = KC + 1
  WRITE(ISYSPU,1004) VC,TI,NBLOCK,KC
300 RETURN
1000 FORMAT (15A4,10X,F6.3)
1001 FORMAT (I10,A10,4E10.4)
1002 FORMAT (1H )
1003 FORMAT (10HCOVARIANCE I10)
1004 FORMAT (7E10.3,F6.3,2I2)
  END
C ***** MXD *****
  SUBROUTINE MXD(M,M1,M2,KD)
  IMPLICIT REAL*8 (A-H,O-Z)
C  MULTIPLIES PAIRS OF MATRICES OF ELEMENTS M1, AND M2 AND
C  STORES AT LQ(M). BEAM ORDER IS M1 FOLLOWED BY M2.
C  R(M)=R(M1)*R(M2), MULTIPLY AND DIVIDE

  INCLUDE 'SWTCH.CCC'
  DIMENSION RX1(2,3),RY1(2,3),RX2(2,3),RY2(2,3),RX(2,3),RY(2,3)
  DIMENSION RW(3),RW1(3),RW2(3),RPQ(4)
  DIMENSION T(49),T1(49),T2(49)

  DO 5 I=1,3
5  RW(I) = 0.
  IF (KD.EQ.3) GO TO 100
C  MULTIPLY MATRICES AS 7 X 7, THEN STORE ACCORDING TO KIND
200 CALL RTRV7(M1,T1,RW1)
205 CALL RTRV7(M2,T2,RW2)
  CALL MXM7(T2,T1,T)
  RW(1) = RW1(1) + RW2(1)
  RW(2) = RW1(2) + RW2(2)
  RW(3) = T(40)
  CALL STOR7(M,T,RW)
  RETURN
100 CONTINUE
  CALL RXY(M1,RX1,RY1,RW1)
  CALL RXY(M2,RX2,RY2,RW2)

```

```

DO 102 I=1,2
DO 101 J=1,2
RX(I,J)=RX2(I,1)*RX1(1,J)+RX2(I,2)*RX1(2,J)
101 RY(I,J)=RY2(I,1)*RY1(1,J)+RY2(I,2)*RY1(2,J)
RX(I,3)=RX2(I,1)*RX1(1,3)+RX2(I,2)*RX1(2,3)+RX2(I,3)
RY(I,3)=RY2(I,1)*RY1(1,3)+RY2(I,2)*RY1(2,3)+RY2(I,3)
102 RW(I)=RW1(I)+RW2(I)
DO 103 I=1,2
RPQ(I) = RX(1,3)*RX(2,I)-RX(2,3)*RX(1,I)
103 RPQ(I+2)= RY(1,3)*RY(2,I)-RY(2,3)*RY(1,I)
RW(3)=RW1(3)+RW2(3)-RX1(1,3)*(RX2(2,3)*RX2(1,1)-RX2(1,3)*RX2(2,1))
1      -RY1(1,3)*(RY2(2,3)*RY2(1,1)-RY2(1,3)*RY2(2,1))
2      -RY1(2,3)*(RY2(2,3)*RY2(1,2)-RY2(1,3)*RY2(2,2))
3      -RX1(2,3)*(RX2(2,3)*RX2(1,2)-RX2(1,3)*RX2(2,2))
CALL STXY(M,RX,RY,RW)
CALL STDAT(M,5,4,4,RPQ)
RETURN
END

```

C ***** MXDIV *****

```

SUBROUTINE MXDIV(V,IN,IM,DET,IDIM)
IMPLICIT REAL*8(A-H,O-Z)

```

C MATRIX DIVISION

C REPLACES B BY (A INVERSE)*B. DESTROYS A.

C A IS N BY N, B IS N BY M. A AND B ARE STORED IN V(N,N+M).

C N MUST NOT EXCEED 4 .

INTEGER PERM,P

DIMENSION PERM(4)

DIMENSION V(IDIM,1)

N=IN

M=MAX0(IM,0)

C

NM1=N-1

NP1=N+1

NPM=N+M

200 DO 201 I=1,N

PERM(I)=I

201 CONTINUE

DETERM=1.0

P=1

202 CONTINUE

IMIN=P+1

C

FIND PIVOT

PIVOT=0.

DO 209 I=P,N

DO 208 J=P,N

SAVE=DABS(V(I,J))

IF (SAVE .LE. PIVOT) GO TO 208

PIVOT=SAVE

IBIG=I

JBIG=J

208 CONTINUE

209 CONTINUE

IF (PIVOT .GT. 0.0) GO TO 210

DETERM=0.0

GO TO 290

C ROW EXCHANGE AND ROW REDUCTION

```

210 PIVOT=V(IBIG,JBIG)
    DETERM=DETERM*PIVOT
    DO 219 J=P,NPM
        SAVE=V(IBIG,J)/PIVOT
        V(IBIG,J)=V(P,J)
        V(P,J)=SAVE
219 CONTINUE
    IF (IBIG .NE. P) DETERM=-DETERM
220 CONTINUE
    IF (P .GE. N) GO TO 250

```

C COLUMN EXCHANGE

```

    IF (JBIG .EQ. P) GO TO 230
    ISAVE=PERM(P)
    PERM(P)=PERM(JBIG)
    PERM(JBIG)=ISAVE
    DO 229 I=1,N
        SAVE=V(I,P)
        V(I,P)=V(I,JBIG)
        V(I,JBIG)=SAVE
229 CONTINUE
    DETERM=-DETERM
230 CONTINUE
    V(P,P)=1.0

```

C BLOCK REDUCTION

```

    DO 239 I=IMIN,N
        DO 238 J=IMIN,NPM
            V(I,J)=V(I,J)-V(I,P)*V(P,J)
238 CONTINUE
        V(I,P)=0.0
239 CONTINUE
    P=P+1
249 GO TO 202

```

C BACK SUBSTITUTION

```

250 CONTINUE
    IF (M .LE. 0) GO TO 290
    DO 259 K=NP1,NPM
        V(N,1)=V(N,K)
        DO 253 P=1,NM1
            I=N-P
            V(I,1)=V(I,K)
            DO 252 J=I,NM1
                V(I,1)=V(I,1)-V(J+1,1)*V(I,J+1)
252 CONTINUE
253 CONTINUE
            DO 256 J=1,N
                P=PERM(J)
                V(P,K)=V(J,1)
256 CONTINUE
259 CONTINUE

```

C

```

290 CONTINUE
    DET=DETERM
    RETURN
    END

```

C ROW EXCHANGE AND ROW REDUCTION

```

210 PIVOT=V(IBIG,JBIG)
    DETERM=DETERM*PIVOT
    DO 219 J=P,NPM
        SAVE=V(IBIG,J)/PIVOT
        V(IBIG,J)=V(P,J)
        V(P,J)=SAVE
219 CONTINUE
    IF (IBIG .NE. P) DETERM=-DETERM
220 CONTINUE
    IF (P .GE. N) GO TO 250

```

C COLUMN EXCHANGE

```

    IF (JBIG .EQ. P) GO TO 230
    ISAVE=PERM(P)
    PERM(P)=PERM(JBIG)
    PERM(JBIG)=ISAVE
    DO 229 I=1,N
        SAVE=V(I,P)
        V(I,P)=V(I,JBIG)
        V(I,JBIG)=SAVE
229 CONTINUE
    DETERM=-DETERM
230 CONTINUE
    V(P,P)=1.0

```

C BLOCK REDUCTION

```

    DO 239 I=IMIN,N
        DO 238 J=IMIN,NPM
            V(I,J)=V(I,J)-V(I,P)*V(P,J)
238 CONTINUE
        V(I,P)=0.0
239 CONTINUE
    P=P+1
249 GO TO 202

```

C BACK SUBSTITUTION

```

250 CONTINUE
    IF (M .LE. 0) GO TO 290
    DO 259 K=NP1,NPM
        V(N,1)=V(N,K)
        DO 253 P=1,NM1
            I=N-P
            V(I,1)=V(I,K)
            DO 252 J=I,NM1
                V(I,1)=V(I,1)-V(J+1,1)*V(I,J+1)
252 CONTINUE
253 CONTINUE
        DO 256 J=1,N
            P=PERM(J)
            V(P,K)=V(J,1)
256 CONTINUE
259 CONTINUE

```

C

```

290 CONTINUE
    DET=DETERM
    RETURN
    END

```

```

C ***** MXM6 *****
      SUBROUTINE MXM6(A,B,C)
      IMPLICIT REAL*8 (A-H,O-Z)
C  MULTIPLES 6 X 6 MATRICES
      DIMENSION A(6,1),B(6,1),C(6,1),CC(6,6)
      DO 100 I=1,6
      DO 90 J=1,6
      SUM = 0.
      DO 80 II=1,6
80    SUM = SUM + A(I,II)*B(II,J)
90    CC(I,J) = SUM
100   CONTINUE
      DO 115 I=1,6
      DO 115 J=1,6
      C(I,J) = CC(I,J)
115   CONTINUE
      RETURN
      END

C ***** MXM7 *****
      SUBROUTINE MXM7(A,B,C)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(7,1),B(7,1),C(7,1),CC(7,7)
C  MULTIPLES 7X7 MATRICES
      DO 100 I=1,7
      DO 90 J=1,7
      SUM=0.
      DO 80 II=1,7
80    SUM=SUM+A(I,II)*B(II,J)
90    CC(I,J)=SUM
100   CONTINUE
      DO 115 I=1,7
      DO 115 J=1,7
      C(I,J)=CC(I,J)
115   CONTINUE
      RETURN
      END

C ***** MXV6 *****
      SUBROUTINE MXV6(T,V,VN)
      IMPLICIT REAL*8 (A-H,O-Z)
C  MULTIPLES A 6 X 6 MATRIX BY A 6 COMPONENT VECTOR
      DIMENSION T(6,6),V(1),VN(1),VV(6)
      DO 10 I=1,6
      VV(I) = 0.
      DO 10 J=1,6
      VV(I) = VV(I) + T(I,J)*V(J)
10    CONTINUE
      DO 15 J=1,6
15    VN(J) = VV(J)
      RETURN
      END

C ***** MXV7 *****
      SUBROUTINE MXV7(T,V,VN)
      IMPLICIT REAL*8 (A-H,O-Z)
C  MULTIPLES VECTOR BY A MATRIX
      DIMENSION T(7,7),V(7),VN(7),VV(7)

```

```

      DO 10 I=1,7
      VV(I) = 0.
      DO 10 J=1,7
      VV(I) = VV(I) + T(I,J)*V(J)
10    CONTINUE
      DO 15 J=1,7
15    VN(J) = VV(J)
      RETURN
      END

C ***** NAMCHK *****
      SUBROUTINE NAMCHK(M)
      IMPLICIT REAL*8 (A-H,O-Z)
C   LOOKS FOR IDENTICAL NAME PREVIOUSLY DEFINED.
C   BLANKS OR NO NAMES ARE NOT CONSIDERED IDENTICAL.
C   INFO OF PREVIOUSLY DEFINED ELEMENTS IS REPLACED BY THAT OF THE
C   LAST DEFINED ONE, EXCEPT FOR INFO(3,I), WHICH CONTAINS NXTM
      INCLUDE 'BINFF.CCC'
      INCLUDE 'STORE.CCC'
      DIMENSION INF(24)
      INTEGER doIBLANK/4H /, doIASTRK/4H* /
C
      NAME=INFF(2,M)
      M1=MIN+1
      DO 120 J=M1,MAX
      INAM = INFF(2,J)
      IF (INAM.EQ.NAME.AND.NAME.NE.doIBLANK) GO TO 105
      GO TO 120
C   DUPLICATE NAME, NOT BLANK HAS BEEN FOUND
C   REPLACE INFO(I,J) WITH THAT OF INFO(I,M), EXCEPT FOR INFO(3,J) WHICH
C   CONTAINS NXTM AND TRA
105   CALL RDINF(INF,M)
      INF(8) = doIASTRK
      INF(9) = doIASTRK
      CALL WRTINF(INF,J)
      RETURN
120   CONTINUE
      RETURN
      END

C ***** NPOL *****
      SUBROUTINE NPOL(M,V,PAR)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C   Q   NPOL   N   J   L   CM   BRHO
C   N = ORDER OF PERTURBATION IN HAMILTONIAN, M=N-1
C   J = 0,1 FOR NORMAL OR SKEW N-POLE LENS
C   L = EFFECTIVE LENGTH
C   CM = TAYLOR EXPANSION COEFFICIENT OF MEDIAN-PLANE EXPRESSION OF FIELD.
C
C   J=0   BY + I*BX = SUM(CM/M!) Z**M      (Z = X+I*Y)
C   J=1   BX - I*BY = SUM(CM/M!) Z**M
C   PARTICLE V(I) = X,PX,Y,PY,O,O,DPP
C   N = 2 QUAD, N=3 SEXT, N=4 OCT, N=5, DEC
C
      COMMON/ML/ML

```

```
INCLUDE 'BINFF.CCC'
```

```
INCLUDE 'DIM.CCC'
```

```
DIMENSION V(1),PAR(1),R(7,7),RW(3),ML(3),PR(3),W(6)
```

```
REAL*8 L,LH
```

```
EQUIVALENCE (PR(1),L),(PR(2),CM),(PR(3),BRHO),(W(1),X),
```

```
1 (W(2),PX),(W(3),Y),(W(4),PY),(W(6),DPP)
```

```
COMPLEX*16 II,Z,DP,Q
```

```
DATA II/(0.,1.D0)/
```

```
C
```

```
NN=M
```

```
DO 5 I=1,3
```

```
5 PR(I) = PAR(I)
```

```
DO 6 I=1,6
```

```
6 W(I) = V(I)
```

```
BR = (1.D0+DPP)*BRHO
```

```
UM = CM/BR
```

```
IF (L.EQ.0.) GO TO 7
```

```
LH = L/2.D0
```

```
X = X + PX*LH
```

```
Y = Y + PY*LH
```

```
UM=L*UM
```

```
7 NP = INFF(4,NN)
```

```
JP = INFF(5,NN)
```

```
MP = NP - 1
```

```
MPM=NP-2
```

```
Z=X+II*Y
```

```
CALL RTRV7(MUNIT,R,RW)
```

```
C
```

```
Q=UM
```

```
IF(JP.EQ.1)Q=II*Q
```

```
IF(MPM.EQ.0) GO TO 12
```

```
DO11 I=1,MPM
```

```
11 Q=Q*Z/I
```

```
12 DP=Q*Z/MP
```

```
R(2,1)=-DREAL(Q)
```

```
R(2,3)=DIMAG(Q)
```

```
DPX=-DREAL(DP)
```

```
DPY=DIMAG(DP)
```

```
C
```

```
40 R(4,3) = -R(2,1)
```

```
R(4,1) = R(2,3)
```

```
V(2) = PX + DPX
```

```
V(4) = PY + DPY
```

```
IF (L.EQ.0) GO TO 41
```

```
V(1) = X + V(2) *LH
```

```
V(3)=Y+V(4)*LH
```

```
C
```

```
41 R(2,6) = -DPX
```

```
R(2,7) = -DPX
```

```
R(4,6) = -DPY
```

```
R(4,7) = -DPY
```

```
R(5,1) = DPX
```

```
R(5,3) = DPY
```

```
RW(2)=-DPX
```

```

      IF (L.EQ.0) GO TO 45
      ML(1) = MEND - 3
      ML(2) = M
      ML(3) = MEND - 3
      CALL DRIFT(MEND-3,LH)
      CALL STOR7(M,R,RW)
      CALL MMM(M,3,ML)
      RETURN
45  CALL STOR7(M,R,RW)
      RETURN
      END

C ***** OPNUM *****
      INTEGER FUNCTION OPNUM(OPNAME)
      IMPLICIT REAL*8 (A-H,O-Z)
C  CHECKS OPNAME AGAINST LIST OF ACCEPTABLE NAMES
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'OPLIST.CCC'
      INCLUDE 'GRR.CCC'
      INTEGER OPNAME

      INTEGER IBLANK/4H /
      KOP = OPNAME

      DECODE(4,8,OPNAME) KOP
8    FORMAT(A3,1X)

C  LOOK FOR MAP INSTRUCTION
      IF (KOP.EQ. 3HMAP) GO TO 6
C  RUN THROUGH LIST
      DO 1 I=1,200
      IF(OPNAME.EQ. LIST(I)) GO TO 2
1    CONTINUE
C  IF OPNAME IS ILLEGAL OP IS SET TO 0 AND ERROR SWITH IS TURNED ON
      OPNUM=0
C  IF OPNAME IS BLANK, IT IS ERROR UNLESS IERR IS ON
      IF(OPNAME.NE.IBLANK) GO TO 1111
      IF (IERR) GO TO 3
1111 ERROR=.TRUE.
4    WRITE(3,5) OPNAME
5    FORMAT (15H *** ERROR *** ,A5,'IS AN ILLEGAL INSTRUCTION.')
      GO TO 3
6    OPNUM = 69
      GO TO 3
C  ASSIGN INDEX NUMBER
2    OPNUM = I
3    CONTINUE
      RETURN
      END

C ***** ORBIT *****
      SUBROUTINE ORBIT(MM)
      IMPLICIT REAL*8 (A-H,O-Z)
C*****NOTE*****      1-7-86
C      FILES ORB5 AND ORB12 ARE IN COMMON BLOCK FILES.
C      FILE ORB12 IS MADE BY CYX. ITS NUMBER IS SET IN BLOCK DATA.
C      FILE ORB5 COMES FROM cyx as well (EC, 12/12/86)

```



```

C      file orb3 contains printout of correction activities
C
C ORBIT PROGRAM USES MICADO TO CALCULATE OPTIMIZED CORRECTORS
C FOR GIVEN ORBIT WITH FIELD ERRORS.
C FORMAT IS
C
C      ORBC KA KB CY FX MON COR DELM DELY
C
C WHERE KA=0 FOR HORIZONTAL, 1 FOR VERTICAL CORRECTION
C FX = NAME OF A FXPT INSTRUCTION DEFINING THE INITIAL ORBIT
C (FX MUST HAVE A 1 IN COLUMN 55)
C CY IS THE NAME OF A CYC INSTRUCTION WITH IDENTICAL BEAM LINE
C (KA OF CYC SHOULD BE 12 OR -12)
C MON IS NAME OF ELEMENTS IN BEAM LINE WHERE DISPLACEMENTS ARE
C ASSUMED TO BE MEASURED
C COR IS NAME OF CORRECTION ELEMENTS IN BEAM LINE
C (THERE SHOULD BE MANY IDENTICAL MON-S AND COR-S IN BEAM LINE)
C DELM = ASSUMED MEASUREMENT ERROR (+-DELM/2)
C DELY = TOLERABLE RANGE OF FINAL RESIDUAL ORBIT ERRORS
C (ITERATIONS WILL STOP WHEN THIS LEVEL IS REACHED)
C ABS(KB) = MAX NO. OF ITERATIONS. IF KB NEGATIVE, ALL INTER-
C MEDIATE ITERATIONS AS WELL AS THE MATRIX RELATING ORBIT
C DISPLACEMENTS TO CORRECTORS ARE PRINTED OUT.
C IF KB POSITIVE, ONLY FIRST AND LAST ITERATIONS ARE PRINTED
C AND MATRIX PRINT IS SUPPRESSED.
      INCLUDE 'CORB.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BMI1.CCC'
      INCLUDE 'AAA.CCC'
      INCLUDE 'FILES.CCC'
      common/seed/nr
      REAL*8 MX,MY,MU(2,84),NU,K(84),L(84),M11,M12
      LOGICAL KPR,KPP
      INTEGER BDAT
C
      PR(A,B,C,D)=A*C+B*D
      nrr=nr
      MCYC = MDAT(MM,1)
      MFXP = MDAT(MM,2)
      KA = INFF(4,MM)
      KB = INFF(5,MM)
      MBML=MDAT(Mfxp,2)
      NDAT=INFF(17,MBML)
      CALL MIFILL(MBML,1,NDAT,NELS1,M11)
      MON = BDAT(MM,3)
      LM1 = LOCC(MON,1,NELS1,M11,NB,NMON)
      M=NMON
      KPR = .FALSE.
      IF (KB.LE.0) KPR = .TRUE.
      ITER = IABS(KB)
      IF (KB.EQ.0) ITER = 1
      KPP = KPR
      IFLAG=KA
      EPS = FDAT(MM,1)
      EPSB = FDAT(MM,2)

```

```

EPSZ = EPS
NACOR = BDAT(MM,4)
LCOR = LOCC(NACOR,1,NELS1,MI1,NX,N)
IF (KA.EQ.1) GO TO 9
NAMONH = MON
NACORH = NACOR
NNH = N
GO TO 26
9  NAMONV = MON
   NACORV = NACOR
   NNV = N
26  CONTINUE
C
   write (orb3, 3)
3  FORMAT(1H1)
   write (orb3, 106)EPSB, EPS
106 FORMAT(1X'DISPERSION DE MESURE ',F10.5/
      .      1X'C.A.C. RESIDUEL %      ',F10.5/)
   write (orb3, 100 )
100 FORMAT(1X'CORRECTEURS'/)
   write (orb3, 2)(NX(I),I=1,N)
2  FORMAT (2I5)
   write (orb3, 4 )
   write (orb3, 101)
101 FORMAT(1X'STATIONS'/)
   write (orb3, 1)(NB(I),I=1,M)
1  FORMAT (2I5)
   write (orb3, 4 )
4  FORMAT(1H0)
   IB=1
   IC=1
   REWIND orb5
   REWIND orb12
   IF(IFLAG.EQ.0) GO TO 10
   PX=0.
   PY=1.D0
   GO TO 11
10  PX=1.D0
   PY=0.
11  READ(ORB5) IEL
   PI=4.D0*DATAN(1.D0)
   ICC=0
   DO 12 I=1, IEL
   READ(ORB5) IE ,NO,NAME,DUM,BY,BX,AY,AX,MY,MX
   READ(ORB12) EX,EXP,EY,EYP,XP,DXP,Y,DYP
   IF(NO.EQ.NB(IB)) GO TO 13
   IF(NO.EQ.NX(IC)) GO TO 14
   GO TO 12
13  IA=1
   IX=IB
   IB=IB+1
   B (IX)=PR(XP, Y,PX,PY)+(RAN(nr)-.5)*EPSB
   AP(IX) = 0.
   GO TO 15
14  IF(ICC.NE.0) GO TO 21

```

```

NX(IC)=NX(IC)+1
ICC=1
IA=2
IX=IC
L      (IX)=DUM(1)
K      (IX)=PR(-DUM(5),DUM(5),PX,PY)
ALPHA  (IX)=PR(AX,AY,PX,PY)
15 BETA(IA,IX)=PR(BX,BY,PX,PY)
MU  (IA,IX)=PR(MX,MY,PX,PY)
IF(NO.EQ.NX(IC)) GO TO 14
GO TO 12
21 ICC=0
IX=IC
IC=IC+1
12 CONTINUE
nr=nrr
NU=PR(MX,MY,PX,PY)*PI
IMAX = MXMIN(B,M,1)
IMIN = MXMIN(B,M,0)
DB = (B(IMAX)-B(IMIN))/1000.D0
write (orb3, 203)DB
203 FORMAT ('0BEFORE CORRECTIONS RMAX-RMIN = 'E13.4)
write (orb3, 102 )
102 FORMAT(' ORBITE AVANT CORRECTION(MM)')
write (orb3, 7)( B(I),I=1,M)
7 FORMAT(7F10.5)
SP=2.D0*DSIN(NU)
DO 118 I=1,M
SBI = DSQRT(BETA(1,I))
DO 118 J=1,N
DM=(MU(1,I)-MU(2,J))*2.D0*PI+NU
IF(MU(2,J).LT.MU(1,I)) DM=DM-2.D0*NU
SB=DSQRT(BETA(2,J))
CM=DCOS(DM)
IF(I.GT.1) GO TO111
NX(J) = NX(J)-1
111 CONTINUE
IF (L(J).EQ.0.) GO TO 116
SM=DSIN(DM)
IF(K(J)) 17,18,19
17 AK=DABS(K(J))
SK=DSQRT(AK)
M11=DSINH(SK*L(J))/SK
M12=(DCOSH(SK*L(J))-1.D0)/AK
GO TO 16
18 M11=L(J)
M12=L(J)*L(J)/2.D0
GO TO 16
19 SK=DSQRT(K(J))
M11=DSIN(SK*L(J))/SK
M12=(1.D0-DCOS(SK*L(J)))/K(J)
16 A(I,J)=DSQRT(BETA(1,I))*(SB*CM*M11+(SM-ALPHA(J)*CM)*M12/SB)/SP
GO TO 118
116 A(I,J) = SBI*SB*CM/SP
118 CONTINUE

```

```

      IF (.NOT.KPP) GO TO 1181
      write (orb3, 4)
      write (orb3, 103 )
103  FORMAT(1X'MATRICE A'/)
      write (orb3, 8)((A(I,J),J=1,N),I=1,M)
      8 FORMAT(6(1X12F10.4)/)
1181  CONTINUE
      DO 22 J=1,ITER
      write (orb3, 4)
      DO 23 I1=1,M
      BB(I1)=B(I1)/1000.D0
      DO 23 I2=1,N
23  AA(I1,I2)=A(I1,I2)
      JB=J
      JTER=J
      CALL MICADO(AA,BB,84,M,N,AP,0.,0.,0.,EPS,JTER,DP,X,NV,R,RHO)
      IF (J.EQ.1.OR.J.EQ.ITER.OR.EPS.LT.EPS7) KPP = .TRUE.
      IF(JTER.LT.JB) RETURN
      IF (KPP) write (orb3, 104 )
104  FORMAT(1X'CORRECTIONS      MRAD      MRAD,MM'/)
      DO 20 I=1,J
      X(I)=X(I)*1000.D0
      XX(I)=X(I)
      NK=NV(I)
      IF(K(NK).NE.0.) XX(I)=X(I)/K(NK)
      IF (KPP) write (orb3, 5) NK,NX(NK),X(I),XX(I)
      IF(IFLAG.EQ.0) TH(NV(I))=-X(I)/1000.D0
      IF(IFLAG.EQ.1) TV(NV(I))= X(I)/1000.D0
20  CONTINUE
      5 FORMAT(2I5,2F10.5)
      DO 24 I=1,M
24  R(I)=R(I)*1000.D0
      IF (.NOT.KPP) GO TO 22
      IMAX = MXMIN(R(1),M,1)
      IMIN = MXMIN(R(1),M,0)
      EPS1=R(IMAX)-R(IMIN)
      EPS1 = EPS1/1000.D0
      write (orb3, 201) J,EPS1
201  FORMAT(1H0,5HAFTER,I3,22H ITERATIONS,RMAX-RMIN=,E13.4)
      write (orb3, 4 )
      write (orb3, 105 )
105  FORMAT(1X'ORBITE CORRIGEE (MM)'/)
      write (orb3, 6) (I,R(I) ,I=1,M)
22  KPP = KPR
      6 FORMAT(7(I5,F10.5))
      RETURN
      END
C***** NBR *****
      SUBROUTINE NBR(L,N)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION L(12)
      N=0
      DO 1 I=1,12
      IF(L(I).EQ.0) RETURN
      IF(L(I).EQ.10H )RETURN

```

```

1 N=I
END

```

```

C***** HTUL *****

```

```

SUBROUTINE HTUL(A,M,N,K,SIG,BETA,M1)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(M1,1)
SIG=0.0

```

```

C

```

```

DO 10 I=K,M
SIG=SIG+(A(I,K))*DBLE(A(I,K))

```

```

10 CONTINUE

```

```

SIG=DSQRT(SIG)

```

```

C

```

```

C ON CHOISIT LE SIGNE CORRECT POUR SIG.

```

```

H=A(K,K)

```

```

IF(H.LT.0.0)SIG=-SIG

```

```

BETA=H + SIG

```

```

A(K,K)=BETA

```

```

BETA=1.D0/(SIG*BETA)

```

```

RETURN

```

```

END

```

```

C***** HTAL *****

```

```

SUBROUTINE HTAL(A,M,N,K,BETA,M1)

```

```

IMPLICIT REAL*8 (A-H,O-Z)

```

```

DIMENSION A(M1,1)

```

```

NC=N-K

```

```

DO 21 J=1,NC

```

```

H=0.0

```

```

DO 20 K1=K,M

```

```

20 H=H+(A(K1,K))*(A(K1,K+J))

```

```

H=BETA*H

```

```

DO 21 K1=K,M

```

```

21 A(K1,K+J)=A(K1,K+J)-A(K1,K)*H

```

```

RETURN

```

```

END

```

```

C***** HTBL *****

```

```

SUBROUTINE HTBL(A,B,M,N,K,BETA,M1)

```

```

IMPLICIT REAL*8 (A-H,O-Z)

```

```

DIMENSION A(M1,1),B(1)

```

```

H=0.

```

```

DO 20 K1=K,M

```

```

20 H=H+(A(K1,K))*(B(K1))

```

```

H=BETA*H

```

```

DO 21 K1=K,M

```

```

21 B(K1)=B(K1)-A(K1,K)*H

```

```

RETURN

```

```

END

```

```

C***** HTRL *****

```

```

SUBROUTINE HTRL(A,B,M,N,K,AUX,M1)

```

```

IMPLICIT REAL*8 (A-H,O-Z)

```

```

DIMENSION A(M1,1),B(1),AUX(1)

```

```

C CALL UZERO(B,1,K)

```

```

DO 10 I=1,K

```

```

B(I)=0.0

```

```

10 CONTINUE

```

```

DO 30 KK=1,K
LV=M-K+KK
KN=N+K-KK+1
KL=K-KK+1

C
C
BETA=-1.D0/(AUX(KN)*A(KL,KL))
CALL HTBL(A,B,M,N,KL,BETA,M1)
30 CONTINUE
RETURN
END

C ***** MXMIN *****
FUNCTION MXMIN (A,N,M)
IMPLICIT REAL*8 (A-H,O-Z)
C IF M=0, MAXMIN=LOWEST INDEX OF MINIMUM ELEMENT IN A
C IF M=1, MAXMIN=LOWEST INDEX OF MAXIMUM ELEMENT IN A
C IF MZ1, MAXMIN=1
DIMENSION A(N)
MXMIN = 1
IF (N.LT.1) RETURN
CURENT=A(1)
DO 10 I=2,N
IF ((M.EQ.0).AND.(A(I).GE.CURENT)) GO TO 10
IF ((M.EQ.1).AND.(A(I).LE.CURENT)) GO TO 10
CURENT=A(I)
MXMIN = I
10 CONTINUE
RETURN
END

C***** HTLS *****
SUBROUTINE HTLS(A,B,M1,M,N,ITER,X,IPI,R,AUX,EPS)
C
C
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(M1,1),B(1),X(1),IPI(1),R(1),AUX(1)
DOUBLE PRECISION G,H,SIG
C
C CALCUL DU PREMIER PIVOT
C
K2=N + 1
PIV=0.0
DO 4 K=1,N
IPI(K)=K
H=0.
G = 0.
DO 2 I=1,M
H=H+(A(I,K))*(A(I,K))
G=G+(A(I,K))*(B(I))
2 CONTINUE
AUX(K)=H
AUX(K2) = G
PIVT = G*G/H
IF(PIVT-PIV)4,4,3
3 PIV = PIVT
KPIV=K
4 K2 = K2 + 1

```

```

C   BOUCLE POUR CHAQUE ITERATION
      DO 21 K=1,ITER
      IF(KPIV.EQ.K)GO TO 8
C   ON ECHANGE LES K ET KPIV SI KPIVPLUS GRAND QUE K
      H=AUX(K)
      AUX(K)=AUX(KPIV)
      AUX(KPIV)=H
      K2=N+K
      K3=N+KPIV
      G = AUX(K2)
      AUX(K2) = AUX(K3)
      AUX(K3) = G
      DO 7 I=1,M
      H=A(I,K)
      A(I,K)=A(I,KPIV)
      A(I,KPIV)=H
      7 CONTINUE
C   CALCUL DE BETA,SIGMA ET UK DANS HTUL
C
      8 CALL HTUL(A,M,N,K,SIG,BETA,M1)
C
C   ON GARDE SIGMA DANS AUX(N+K)
      J=N+K
      AUX(J)=-SIG
      IP=IPI(KPIV)
      IPI(KPIV)=IPI(K)
      IPI(K)=IP
      IF(K.EQ.N) GO TO 13
C
C   TRANSFORMATION DE A DANS HTAL
C
      CALL HTAL(A,M,N,K,BETA,M1)
C
C   TRANSFORMATION DE B DANS HTBL
C
      13 CALL HTBL(A,B,M,N,K,BETA,M1)
C
C   RECHERCHE DU PIVOT (K+1)
C
      AUX(K)=DSQRT(PIV)
      IF(K.EQ.N) GO TO 11
      PIV=0.0
      KPIV = K + 1
      J1 = KPIV
      K2=N + J1
      DO 18 J=J1,N
      H=AUX(J)-(A(K,J))*(A(K,J))
      AUX(J)=H
      G=AUX(K2)-(A(K,J))*(B(K))
      AUX(K2) = G
      PIVT = G*G/H
      IF(PIVT.LT.PIV)GO TO 18
      KPIV=J
      PIV=PIVT
      18 K2 = K2 + 1

```

```

C
C  CALCUL DES X
C
11  X(K)=B(K)/AUX(N+K)
    IF(K.EQ.1)GO TO 27
    DO 15 I=2,K
      KK=K-I+1
      X(KK)=B(KK)
      KI=KK+1
    DO 25 J=KI,K
25  X(KK)=X(KK)-A(KK,J)*X(J)
    X(KK)=X(KK)/AUX(N+KK)
15  CONTINUE
C 27 CALL UCOPY(B,R,M)
27  CONTINUE
    DO 29 III=1,M
      R(III) = B(III)
29  CONTINUE
C
C  CALCUL DU VECTEUR RESIDUEL DANS HTRL
    CALL HTRL(A,R,M,N,K,AUX,M1)
C  IMAX=MAXFZE(R(1),M)
C  IMIN=MINFZE(R(1),M)
    IMAX = MXMIN(R(1),M,1)
    IMIN = MXMIN(R(1),M,0)
    EPS1=R(IMAX)-R(IMIN)
    GOTO 1000
C  PRINT OUT OF INTERMEDIATE CORRECTOR STRENGTHS/ORBITS
500. WRITE(3,6000) K,R(IMAX),IMAX,R(IMIN),IMIN
      WRITE(3,6002) (IX,IPI(IX),X(IX),IX=1,K,1)
      WRITE(3,6004)
      WRITE(3,6006) (IX,R(IX),IX=1,M,1)
1000 CONTINUE
    IF(EPS1.GT.EPS)GO TO 21
    GO TO 202
21  CONTINUE
    RETURN
202  EPS=EPS1
    ITER=K
    RETURN
6000 FORMAT(5X,15HNO. CORRECTORS=,I5/
. 5X,11HMAX. ORBIT=,3PF8.3,13HMM AT MONITOR,I5/
. 5X,11HMIN. ORBIT=,3PF8.3,13HMM AT MONITOR,I5//
. 5X,10HCORRECTORS/
. 5X,4H I ,2X,5H NAME,2X,8HSTRENGTH/
. 5X,13X,11HMICRO. RAD./)
6002 FORMAT(5X,I4,2X,I5,2X,6PF11.6)
6004 FORMAT(5X,31HORBIT DISPLACEMENTS AT MONITORS/
. 5X,4H I ,2X,10HMONITOR(I)/
. 5X,6X,3H MM/)
6006 FORMAT(5X,I4,3PF10.5)
    END
C***** PARSR *****
    SUBROUTINE PARSR (THET,ERR)
    IMPLICIT REAL*8(A-H,O-Z)

```



```

C  FUNCTION EVALUATION FOR COLLINS STRAIGHT SECTION DESIGN
      INCLUDE 'AGS.CCC'
      INCLUDE 'STR.CCC'

C
C
      REAL KL

C
      S = DSIN(THET)
      C = DCOS(THET)
      SH = (DEXP(THET) - DEXP(-THET))/2.00
      CS = (DEXP(THET) + DEXP(-THET))/2.00
      X = C*SH + S*CH
      Y = C*SH - S*CH
      Z = C*CH
      W = S*SH
      KL = 2.00*(ALPHA*SMUD - W)/X
      RK = (KL*W - Y)/(GAM*SMUD)
      BL = KL*RK
      BK = 1.00/RK
      LL = (KL*Y + 2.00*(Z - CMUD))/(2.00*GAM*SMUD)
      LK = RHO*(BK*BK)
      LM = THET/BK
      A = BQ/(LK*BZ)
      XI = (1.00 + ALPHA*LL/BETA)**2 + (LL/BETA)**2
      XQ = AVAC*DSQRT(BETA*XI/BMAX)
      ERR = XQ - A
      END

C ***** PINTF *****
      FUNCTION PINTF(PEXTI,I)
      IMPLICIT REAL*8 (A-H,O-Z)

CC      CALCULATES THE INTERNAL PARAMETER VALUE PINTF CORRESPONDING
CC      TO THE EXTERNAL VALUE PEXTI FOR PARAMETER I.
CC
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON ,UNIT / ISYSRD ,ISYSWR ,ISYSPU
      INCLUDE 'CONST.CCC'

      BIG = +PI02
      SMALL=-PI02
      IGO = LCODE(I)
      GO TO (100,200,300,400),IGO
C—      IGO = 1  MEANS NO LIMITS
      100 PINTF = PEXTI
      GO TO 300
      200 CONTINUE
      300 CONTINUE
C—      IGO = 4  MEANS THERE ARE TWO LIMITS
      400 ALIMI = ALIM(I)
      BLIMI = BLIM(I)
      IF (PEXTI-ALIMI) 440,500,460
      440 A = SMALL
      450 PINTF = A
      PEXTI = ALIMI + 0.500* (BLIMI-ALIMI) *(DSIN(A) +1.000)
      LIMSET=1
C      WRITE (ISYSWR,241) I

```

```

      GO TO 800
460 IF (BLIMI-PEXTI) 470,520,480
470 A = BIG
      GO TO 450
480 YY=2.0D0*(PEXTI-ALIMI)/(BLIMI-ALIMI) - 1.0D0
      PINTF = DATAN(YY/DSQRT(1.0D0- YY**2) )
      GO TO 800
500 PINTF = SMALL
      GO TO 800
520 PINTF = BIG
800 RETURN
241 FORMAT ( 19H WARNING - VARIABLE I3, 46H HAS BEFN BROUGHT BACK INS
1IDE LIMITS BY PINTF.)
      END
C*****PBML*****
      SUBROUTINE PBML(M)
C PRINT NAMED BEAM ELEMENTS WITH LENGTHS AND ACCUMULATED LENGTHS.
      IMPLICIT REAL*8 (A-H,O-Z)

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      INCLUDE 'BMI1.CCC'

C
C
C
      NBCD = 1
      DO 100 I=1,NBCD
      NN = MDAT(M,I)
      MNM = MNAME(NN)
      NDAT = INFF(17,NN)
      CALL MIFILL(NN,1,NDAT,NELS1,M11)

C
      WRITE (3,1) MNM
      ELT = 0.

C
      DO 110 J=1,NELS1
      JJ = M11(J)
      MNAM = MNAME(JJ)
      EL = FDAT(JJ,1)
      ELT = ELT + EL
      WRITE (3,2) MNAM,EL,ELT
110 CONTINUE
100 CONTINUE
C
1 FORMAT (1H1,5X,'ELEMENTS THROUGH BEAM LINE ',A5//6X,'NAME',5X,
1 ' LENGTH',6X,' ACC.LENGTH')
2 FORMAT (6X,A5,2X,F12.8,2X,F12.8)
      RETURN
      END

```

File: SYNNOPL.FOR;2

C***** PRNT *****

```

SUBROUTINE PRNT(M)
  IMPLICIT REAL*8 (A-H,O-Z)

C
C PRINTS THE KA-TH PARAMETER OF TYPE KB FOR SPECIFIED ELEMENTS.
C DEFAULTS ARE PARAMETER 1 OF TYPE 5
C TYPES ARE
C     1  INPUT FL. PT. (LFL)
C     2  INPUT BCD      (LBC)
C     3  INPUT INTEGER (LIN)
C     4  NULL
C     5  FL. PT.  (LQ STORAGE)
C     6  FL. PT. STORAGE (SAME AS 1)
C     7  FL. PT. LQ2 STORAGE
C     8  FL. PT. LQ3 STORAGE

  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  INCLUDE 'BMI.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))

C
  DIMENSION VAL(100),IVAL(100),NAMTYPE(8),NAM(100)
  DIMENSION NMTYP1(2),NMTYP6(2)
  EQUIVALENCE (VAL,IVAL)
  INTEGER BDAT,ELNUM

C
  DATA LIM1,LIMF,IBLNK/10,5,' '/
  DATA NAMTYPE/' ',3HBCD,3HINT,' ',2HLQ,' ',
1      3HLQ2,3HLQ3/
  DATA NMTYP1/3HFL ,2HPT/
  DATA NMTYP6/4HSTOR,3HAGE/

C
C RETRIEVE ELEMENTS. PARAMETER AND TYPE. SET DEFAULTS.
  NUM = INFF(17,M)
  KNDX = INFF(4,M)
  KTYPE = INFF(5,M)
  IF (KNDX.EQ.0) KNDX = 1
  IF (KTYPE.EQ.0) KTYPE = 5
  J = 0
  DO 10 I=1,NUM
    NAME = BDAT(M,I)
    IF (NAME.EQ.IBLNK) GO TO 10
    J = J + 1
    NAM(J) = NAME
    MI(J) = ELNUM(NAME)
10  CONTINUE
    NUM = J

C PRINT HEADING
  IF (KTYPE.NE.1.AND.KTYPE.NE.6)
1  WRITE (3,100) KNDX,KTYPE,NAMTYPE(KTYPE)
  IF (KTYPE.EQ.1) WRITE (3,101) KNDX,KTYPE,NMTYP1
  IF (KTYPE.EQ.6) WRITE (3,102) KNDX,KTYPE,NMTYP1(1),NMTYP6

C
  DO 18 I=1,NUM

```

```

      NK = MI(I)
      GO TO (15,12,13,14,15,15,15,15) KTYPE
12   IVAL(I) = BDAT(NK,KNDX)
      GO TO 18
13   IVAL(I) = IDAT(NK,KNDX)
      GO TO 18
14   RETURN
15   CALL DATA (NK,KTYPE,KNDX,1,VAL(I))
18   CONTINUE

```

C

C WRITE NAMES OF ELEMENTS WITH VALUES UNDERNEATH, 10 TO A LINE.

```

      NM = 0
      LIM = LIMF
      IF (KTYPE.EQ.2.OR.KTYPE.EQ.3) LIM = LIM1
20   I1 = NM + 1
      NM = NM + LIM
      IF (NM.GT.NUM) NM = NUM
      IF (KTYPE.EQ.2.OR.KTYPE.EQ.3) WRITE (3,105) (NAM(I),I=I1,NM)
      IF (KTYPE.EQ.1.OR.KTYPE.GT.3) WRITE (3,109) (NAM(I),I=I1,NM)
      GO TO (21,22,23,24,21,21,21,21) KTYPE
21   WRITE (3,111) (VAL(I),I=I1,NM)
      GO TO 25
22   WRITE (3,107) (IVAL(I),I=I1,NM)
      GO TO 25
23   WRITE (3,108) (IVAL(I),I=I1,NM)
      GO TO 25
24   RETURN
25   IF(NM.NE.NUM) GO TO 20
      WRITE (3,110)
      RETURN

```

C

```

100  FORMAT(1X,130(1H-)/2X,'VARIABLE',I3,1X,'OF TYPE',I2,1X,'(',
1    A4,')')
101  FORMAT(1X,130(1H-)/2X,'VARIABLE',I3,1X,'OF TYPE',I2,1X,'(',
1    2A4,')')
102  FORMAT(1X,130(1H-)/2X,'VARIABLE',I3,1X,'OF TYPE',I2,1X,'(',
1    3A4,')')
105  FORMAT(/1X,10(7XA5))
106  FORMAT(2X,10(F12.6))
107  FORMAT(10(8X,A4))
108  FORMAT(10(2X,I10))
109  FORMAT(8X,5(A5,15X))
110  FORMAT(1X,130(1H-))
111  FORMAT (2X,5(F18.9,2X))
      END

```

C ***** PRNTAB *****

```

      SUBROUTINE PRNTAB(M)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION ISTORE(1)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      EQUIVALENCE (ISTORE(1),STORE(1))
      DIMENSION NAM(10),NCG(10),F(10),IVAR(2)
      EQUIVALENCE (FVAR,IVAR(1))

```

```

      INTEGER doITAB/'TAB'/',doIBLNK/' '/,doIAST/'*'/
      INTEGER ELNUM
      LOGICAL PRES

C  IF M IS INDEX OF TAB INSTRUCTION, PRINT THAT ONE ONLY
      PRES = .FALSE.
      IF (INFF(1,M).NE.doITAB) GO TO 200
C  CALLED FROM TABULATE WHEN VECTOR FILLED
      PRES = .TRUE.
      IK = 1
      NAM(1) = INFF(2,M)
      GO TO 205
200  CONTINUE
      KA = INFF(4,M)
      IK = KA
C  PICK UP TAB NAMES
      CALL DATA(M,2,1,KA,NAM)
205  CONTINUE
      DO 220 I=1,IK
C  GET INDEX OF TABLE NAME AND NUMBER OF TABLE ENTRIES
      INDX = ELNUM(NAM(I))
      NUM = INFF(4,INDX)
C  GET NAMES FOR COLUMN HEADINGS AND PRINT THEM
      CALL DATA(INDX,2,1,NUM,NCOL)
      WRITE (3,1) (NCOL(J),J=1,NUM)
C  GET CURSOR. COMPUTE NUMBER OF ROWS TO BE PRINTED
      LQ = INFF(10,INDX)
      FVAR=STORE(LQ)
      LC=IVAR(2)
      NC = (LC-2)/NUM
      IC = 2
      DO 210 J=1,NC
      CALL DATA(INDX,5,IC,NUM,F)
      IAST = doIBLNK
      NK =0
      DO 209 K=1,NUM
209  IF (F(K).EQ.0.) NK = NK + 1
      IF (NK.EQ.NUM) IAST = doIAST
      WRITE (3,2) IAST,(F(K),K=1,NUM)
      IC = IC + NUM
210  CONTINUE
C  RESET CURSOR
      IVAR(2)=2
      STORE(LQ)=FVAR
220  CONTINUE
C  IF CALLED FROM TAB, RESTORE PAGE.
      IF (PRES) WRITE (3,100)
100  FORMAT (1H1)
      RETURN
1  FORMAT (1H1,6X,10(7X,A5)//)
2  FORMAT (4X,A1,2X,10F12.4)
      END
C ***** PRINTV *****
      SUBROUTINE PRINTV(M,K)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

INCLUDE 'BINFF.CCC'
COMMON/SWCH/BEND,MSIZE,CYCSWT,VCSW,PV,MSSW,VPR,MHS,MODS
INTEGER ELNUM,BDAT
LOGICAL CYCSWT,MSSW,VCSW,PV,VPR,MHS,MODS
LOGICAL BEGIN
DIMENSION NUM(7)
DIMENSION NAM(7),V(10,7),IH(12),CV(70)
DOUBLE PRECISION NAM
EQUIVALENCE (CV(1),V(1,1))
DATA IH /1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,2H10,
12H11,2H12/
INTEGER do1SNGL/4HSNGL/,do1CPLX/4HCPLX/
DATA IM,IB/1H-,1H /
BEGIN = .TRUE.
JJ = 0
DO 100 IJ=1,K
IF (PV) GO TO 13
NM = BDAT(M,IJ)
M1 = ELNUM(NM)
GO TO 14
13 NM = MNAME(M)
M1=M
14 NCOL = INFF(23,M1)
NROW = INFF(22,M1)
NTYPE= INFF(21,M1)
LOC = 1
IF (.NOT.VCSW) GO TO 15
NCOL = 1
DECODE(4,6,NM)NUMB
LOC = (NUMB-1)*NROW+1
15 IF (NTYPE.NE.do1SNGL.AND.JJ.NE.0) GO TO 21
IF (NCOL.GT.7) NCOL=7
ISUB=JJ+NCOL
IF (ISUB.GT.7) GO TO 21
IF (NTYPE.NE.do1SNGL) NCOL=1
IF (NROW.NE.NROW1.AND..NOT.BEGIN) GO TO 21
DO 20 JK =1,NCOL
JJ = JJ + 1
JL = JJ*2 -1
NAM(JL) = NM
JM = IB
JN = IB
IF(NCOL.LE.1) GO TO 17
JM = IM
JN = IH(JK)
17 ENCODE(7,7,NAM(JL))NM,JM,JN
IF(NTYPE.EQ.do1SNGL) GO TO 16
C COMPLEX OR DOUBLE PRECISION VECTOR
NC = NROW*2
CALL DATA(M1,5,LOC,NC,CV(1))
GO TO 20
16 CALL DATA(M1,5,LOC,NROW,V(1,JJ))
LOC = LOC + NROW
20 CONTINUE
IF (NTYPE.NE.do1SNGL) GO TO 35

```

```

      BEGIN = .FALSE.
      IF (ISUB.EQ.7.OR.IJ.EQ.K) GO TO 22
      NROW1 = NROW
      GO TO 100
21    ASSIGN 15 TO IRET
      NR = NROW1
      GO TO 25
22    ASSIGN 100 TO IRET
      NR = NROW
C    TIME TO PRINT
25    WRITE (3,1)
      J2 = JJ*2
      CALL DASH
      WRITE (3,4) (NAM(J),J=1,J2)
      DO 30 L=1,NR
      WRITE (3,5)(V(L,J),J=1,JJ)
30    CONTINUE
      GO TO 40
C    PRINT COMPLEX OR DOUBLE PRECISION VECTORS
35    WRITE (3,1)
      CALL DASH
      IF (NTYPE.EQ.doICPLX) GO TO 37
      WRITE (3,9) NAM(1)
      GO TO 38
37    WRITE (3,8) NAM(1)
38    WRITE (3,10) (CV(J),J=1,NC)
      ASSIGN 100 TO IRET
40    BEGIN = .TRUE.
      JJ = 0
      GO TO IRET,(15,100)
100  CONTINUE
      CALL DASH
1    FORMAT (//)
4    FORMAT (4X,7(9X,A4,A3))
5    FORMAT (4X,7(2X,F14.8))
6    FORMAT (1X,I1,2X)
7    FORMAT (A4,A1,A2,1X)
8    FORMAT (16H COMPLEX VECTOR ,A5)
9    FORMAT (25H DOUBLE PRECISION VECTOR ,A5)
10   FORMAT (30X,2(E14.8,5X))
      RETURN
      END
C ***** PRNTV7 *****
      SUBROUTINE PRNTV7(M,K)
      IMPLICIT REAL*8 (A-H,O-Z)

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      INCLUDE 'SWTCH.CCC'

      LOGICAL PH
      DIMENSION V(7)

```

```

INTEGER ELNUM,BDAT
INTEGER doISNGL/'SNGL'/

```

```

C PV IS LOGICAL VARIABLE THAT TELLS PRINTV IT HAS BEEN CALLED BY PRNTV7
C AND SO DOES NOT HAVE LIST OF DIFFERENT VECTORS TO PRINT.
C PH CONTROLS COLUMN HEADING. IF PRINTV WAS CALLED, NEW HEADING
C IS NEEDED.

```

```

    PH = .FALSE.
    CALL DASH
    DO 100 I=1,K
    PV=.FALSE.
    LOC = 1
    NM = BDAT(M,I)
    M1 = ELNUM(NM)
    NROW = INFF(22,M1)
    NTYPE=INFF(21,M1)

```

```

C IF VECTOR IS NOT 7 ROW OR IF COMPLEX OR DOUBLE,CALL OTHER PRINT ROUT.

```

```

    IF (NROW.EQ.7.AND.NTYPE.EQ.doISNGL) GO TO 20
    PV = .TRUE.
    CALL PRINTV(M1,1)
    PV=.FALSE.
    PH = .FALSE.
    GO TO 100

```

```

C PARTICLE VECTOR WITH 7 ROWS

```

```

20  NCOL = INFF(23,M1)
    LOC = 1
    IF (.NOT.VCSW) GO TO 25
    DECODE(4,2,NM)NUM

    LOC = (NUM-1)*7+1
    NCOL = 1
25  IF (.NOT.PH) WRITE (3,10)
    DO 30 J=1,NCOL
    CALL DATA (M1,5,LOC,7,V)
28  JJ = J
    IF (VCSW) JJ=NUM
    WRITE (3,13) NM,JJ,(V(L),L=1,7)
29  LOC = LOC + 7
30  CONTINUE
    PH = .TRUE.
100 CONTINUE
    CALL DASH
    WRITE (3,14)
    RETURN
2   FORMAT (1X,I1,2X)
10  FORMAT (//17H VECTOR X,10X,2HDX,9X,1HY,10X,2HDY,9X,1HS,9X,
14HDP/P)
12  FORMAT (5X,A4,4X,7(F10.5,1X))
13  FORMAT (5X,A4,I2,2X,7(F10.5,1X))
14  FORMAT (///1H )
    END

```

```

C***** PWNSKY ,*****
SUBROUTINE PWNSKY(IJOE)
IMPLICIT REAL*8 (A-H,O-Z)

```



```

COMMON/IBSDAT/ A(40),BTX(512),BTY(512),APX(512),APY(512),APP(512),
1          AP(512),AA(512),BB(512),CC(512),BPAR(10),SS(512)

C
  PI=DACOS(-1.D0)
  WRITE(3,901) IJOE
901  FORMAT(13)
  AMAX=IJOE-1
  IMAX=AMAX
  DO 99 JOE=1,IJOE
  WRITE(3,7001) SS(JOE),BTX(JOE),APX(JOE),BTY(JOE),
CAPY(JOE),AP(JOE),APP(JOE)
99  CONTINUE
  DO 91 JOE=1,IMAX
  BTX(JOE)=(BTX(JOE+1)+BTX(JOE))* .5D0
  BTY(JOE)=(BTY(JOE+1)+BTY(JOE))* .5D0
  APX(JOE)=(APX(JOE+1)+APX(JOE))* .5D0
  APY(JOE)=(APY(JOE+1)+APY(JOE))* .5D0
  APP(JOE)=(APP(JOE+1)+APP(JOE))* .5D0
  AP(JOE)=(AP(JOE+1)+AP(JOE))* .5D0
91  CONTINUE
200 CONTINUE
C
300 DO 310 K=2,7
  BPAR(K)=A(K)
310 CONTINUE
350 GM=BPAR(2)
  EXN=BPAR(3)
  EYN=BPAR(4)
  SPN=BPAR(5)
  SL=BPAR(6)
  PNB=BPAR(7)
  EX=EXN/GM
  EY=EYN/GM
  SP=SPN/GM
  DO 400 K=1,IJOE
  CHI=APP(K)+AP(K)*APX(K)/BTX(K)
  AA(K)=AP(K)**2/(EX*BTX(K))+BTX(K)/EX*CHI**2+2.D0/SP**2
  AA(K)=AA(K)*GM**2
  BB(K)=(BTX(K)/EX+BTY(K)/EY)*(AP(K)**2/(EX*BTX(K))+2.D0/SP**2)
  BB(K)=GM**2*(BB(K)+BTX(K)*BTY(K)/EX/EY*CHI**2)
  CC(K)=AP(K)**2/BTX(K)/EX+2.D0/SP**2
  CC(K)=CC(K)*BTX(K)*BTY(K)/EX/EY*GM**2
  WRITE(3,7002) K,AA(K),BB(K),CC(K)
400 CONTINUE
  SY=0.
  SX=0.
  SLG=0.
  ACM=6.03D-30/(GM*SL*SPN*EXN*EYN)*PNB
  ACM=3.D+8*ACM
  DO 600 K=1,IMAX
  CHI=APP(K)+AP(K)*APX(K)/BTX(K)
  CFL=2.D0*(GM/SP)**2*ACM
  CFX=GM**2/EX*(AP(K)**2/BTX(K)+BTX(K)*CHI**2)*ACM
  CFY=BTY(K)/EY*ACM
  CALL TC(1,1.D0,AA(K),BB(K),CC(K),1.D0,TI)

```

```

TINX=CFX*TI
TINL=CFL*TI
VC=3.D0*EY/BTY(K)
CALL TC(2,VC,AA(K),BB(K),CC(K),1.D0,TI)
TINY=CFY*TI
FJOE=DABS((SS(K+1)-SS(K))/(SS(IJOE)-SS(1)))
SY=SY+TINY*FJOE
SX=SX+TINX*FJOE
SLG=SLG+TINL*FJOE
600  CONTINUE
      WRITE(3,7003) SY,SX,SLG
1000  RETURN
7001  FORMAT(F7.3,2X,6E15.6)
7002  FORMAT(I5,3E14.5)
7003  FORMAT(//10X,3E14.5///)
      END
C***** TC *****
      SUBROUTINE TC(IP,VC,AO,BO,CO,CC,TI)
      IMPLICIT REAL*8 (A-H,O-Z)
      IF(CC.EQ.0.) CC=CO
      H=(CO/CC)**(1.D0/3.D0)
      BB=BO/H**2
      AA=AO/H
      AAL=AA
      BBL=1.D0
      IF(IP.EQ.1) GO TO 10
      AAL=-AA/2.D0
      BBL=(BO-VC*CO)/H**2
      BBLA=BBL/AAL
      XML=DABS(BBLA)
10    BBAA=BB/AA
      CCBB=CC/BB
      XM=DMAX1(AA,BBAA,CCBB)
      IF(IP.EQ.2) XM=DMAX1(XM,XML)
      XM=1.D+3*XM
      DO 200 K=1,20
      S=AAL/XM**2
      CR=0.5D0 *(K-1)
      CALL IINTT(IP,BBL,AA,BB,CC,XM,CR,S1,NMAX)
      Z=(S+S1)/H
      IF(K.EQ.1) GO TO 100
      ZSO=S/SO-1.D0
      TEST=DABS(ZSO)
      IF(TEST.LT.0.01D0) GO TO 300
100   SO=Z
200   CONTINUE
300   WRITE(3,7002) NMAX,Z
      TI=Z
      RETURN
7002  FORMAT(I10,5X,E14.5)
      END
C***** IINTT *****
      SUBROUTINE IINTT(IP,BBL,A,B,C,XM,CR,S,NC)
      IMPLICIT REAL*8 (A-H,O-Z)
      S=0.

```

```

NC=0
X=XM
CALL FNT(IP,BBL,A,B,C,X,FO,F1,F2)
CRFO=CR*FO/F1
DXFO=DX*FO
DX=DABS(CRFO)
CA=DABS(DXFO)
CB=FO
PO=0.
100  X=X-DX
      IF(X.GT.0.) CALL FNT(IP,BBL,A,B,C,X,PO,P1,P2)
      IF(X.LE.0.) DX=DX+X
      S=S+DX*0.5D0*(PO+FO)
      IF(X.LE.0.) RETURN
      FO=PO
      NC=NC+1
      CAPO=CA/PO
      POCB=PO/CB
      DX=DABS(CAPO)*DSQRT(ABS(POCB))
      IF(NC.GT.50000) RETURN
      GO TO 100
7001  FORMAT(5E14.5)
      END
C***** FNT *****
      SUBROUTINE FNT(IP,BBL,A,B,C,X,FO,F1,F2)
      IMPLICIT REAL*8 (A-H,O-Z)
      T=DSQRT(X)
      AL=A
      BL=B
      IF(IP.EQ.2) AL=-A/2.D0
      IF(IP.EQ.2) BL=BBL
      HO=T*(2.D0*AL*X+BL)
      H1=(3.D0*AL*X+0.5D0*BL)/T
      H2=(1.5D0*AL-0.25D0*BL/X)/T
      T=X**3+A*X**2+B*X+C
      U=3.D0*X**2+2.D0*A*X+B
      V=6.D0*X+2.D0*A
      GO=T*(-1.5D0)
      G1=-1.5D0*T*(-2.5D0)*U
      G2=15.D0/4.D0*T*(-3.5D0)*U**2-1.5D0*T*(-2.5D0)*V
      FO=HO*GO
      F1=H1*GO+HO*G1
      F2=H2*GO+2.D0*H1*G1+HO*G2
      RETURN
      END
C***** QUEX *****
      SUBROUTINE QUEX (X,DQX)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'GRR.CCC'

      COMMON/QUE/V1,V2,VV1,VV2,G,CQXZ,CQYZ,CQX,CQY,M1,M2,P1,P2,MS,ME
1,K1,K2
      INTEGER P1,P2
      IF(IERR) RETURN
      V1 = X

```

```

C      CALL GRTB( 1,V2,7,QUEY)
      DQX = (CQX - CQXZ) **2 + G*G
      RETURN
      END

C ***** QUEY *****
      SUBROUTINE QUEY (Y,DQY)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/QUE/V1,V2,VV1,VV2,G,CQXZ,CQYZ,CQX,CQY,M1,M2,P1,P2,MS,ME
      1,K1,K2
      INCLUDE 'GRR.CCC'

      INTEGER P1,P2
      DIMENSION RX(2,3) ,RY(2,3),RW(3)
      DIMENSION RXRY(12),V(7),BXBY(20)
      DIMENSION BX(9),BY(9)
      IF(IERR) RETURN
      V2 = Y
      CALL REPFLT (M1,P1,V1)
      CALL REPFLT (M2,P2,V2)
      CALL DOIT(MS)
      IF (BSW) GO TO 3
      IF(VSW) GO TO 4
      IF (RSW) GO TO 8
      CALL RXY(ME,RX,RY,RW)
      CQX = (RX(1,1) + RX(2,2)) * .5
      CQY = (RY(1,1) + RY(2,2)) * .5
      GO TO 5
3      CALL BET(ME,BXBY(1),BXBY(11),0)
      CQX = BXBY(K1)
      CQY = BXBY(K2)
      GO TO 6
4      CALL DATA (ME,5,1,7,V)
      CQX = V(K1)
      CQY = V(K2)
      GO TO 6
8      CALL RXY(ME,RXRY(1),RXRY(7),RW)
      CQX = RXRY(K1)
      CQY = RXRY(K2)
6      CQX = CQX/100.D0
      CQY=CQY/100.D0
5      DQY = CQY - CQYZ
      G = DQY
      RETURN
      END

C ***** QFUNCT *****
      SUBROUTINE QFUNCT(XK,XKPRIM,Q)
      IMPLICIT REAL*8 (A-H,O-Z)
      XKPRIM=DSQRT(1.D0-XK**2)
      A=DMAX1(DABS(XK),XKPRIM)
      EPSI=(1.D0-A)/(2.D0*A+4.D0*DSQRT(A)+2.D0)
      E=EPSI**4
      Q=EPSI+EPSI*E*(2.D0+E*(15.D0+E*150.D0))
      IF(XKPRIM.LT.A) Q=DEXP(9.869604401089359D0/DLOG(Q))
      RETURN
      END

```

```

C ***** RAZZIA *****
      SUBROUTINE RAZZIA(YNEW,PNEW)
      IMPLICIT REAL*8 (A-H,O-Z)
CC      CALLED ONLY BY SIMPLEX (AND IMPROV) TO ADD A NEW POINT
CC      AND REMOVE AN OLD ONE FROM THE CURRENT SIMPLEX, AND GET THE
CC      ESTIMATED DISTANCE TO MINIMUM.
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON/CASC/ JH, JL, Y(16)
      COMMON /SIMVEC/ P(15,16),PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15)
      DIMENSION PNEW(15)
      DO 10 I=1,NPAR
10    P(I,JH)=PNEW(I)
      Y(JH)=YNEW
      IF(YNEW.GE.AMIN) GO TO 18
      DO 15 I=1,NPAR
15    X(I)=PNEW(I)
      CALL INTOEX(X)
      AMIN=YNEW
      JL=JH
18    CONTINUE
      JH=1
      NPARP1=NPAR+1
20    DO 25 J=2,NPARP1
      IF (Y(J) .GT. Y(JH)) JH = J
25    CONTINUE
      SIGMA = Y(JH) - Y(JL)
      IF (SIGMA .LE. 0.D0) GO TO 45
      US = 1.0D0/SIGMA
      DO 35 I= 1, NPAR
      PBIG = P(I,1)
      PLIT = PBIG
      DO 30 J= 2, NPARP1
      IF (P(I,J) .LT. PLIT) PLIT = P(I,J)
      IF (P(I,J) .LT. PLIT) PLIT = P(I,J)
30    CONTINUE
      DIRIN(I) = PBIG - PLIT
      IF (ITAU .LT. 1 ) V(I,I) = 0.5D0*(V(I,I) +US*DIRIN(I)**2)
35    CONTINUE
40    RETURN
45    WRITE (ISYSWR, 1000) NPAR
      GO TO 40
1000 FORMAT ( 59H0***** FUNCTION VALUE DOES NOT SEEM TO DEPEND ON ANY
      10F THE I3, 20H VARIABLE PARAMETERS/15X 58HVERIFY THAT STEP SIZ
      2ES ARE BIG ENOUGH AND CHECK FCN LOGIC./1X 81(1H*)/1X 81(1H*)///)
      END
C ***** RDINF *****
      SUBROUTINE RDINF(INF,M)
      IMPLICIT REAL*8 (A-H,O-Z)
C READS ONE BLOCK OF INFF(M) INTO INF
      INCLUDE 'BINFF.CCC'
      DIMENSION INF(1)

```

```

      DO 20 I=1,24
20  INF(I) = INFF(I,M)
      RETURN
      END
C ***** RDINST *****
      SUBROUTINE RDINST
      IMPLICIT REAL*8 (A-H,O-Z)
C PROCESSES FIRST CARD OF NEW INSTRUCTION WHICH IS CURRENTLY IN CARD (8)
C BY STORING NAME,OPNAME.
      INCLUDE 'INSTR.CCC'
      INCLUDE 'STORE.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'DIM.CCC'
      INCLUDE 'FILES.CCC'

      INTEGER OPNUM
      DIMENSION IC(24)
      EQUIVALENCE( IC(1),OPNAME )
C
      IF (MIN .GE. 1) GO TO 7
      WRITE(3, 200)
200  FORMAT (6H0*****,'ELEMENTS OVERFLOWED. INCREASE INFF ARRAY AND ',
1    'MAX. ')
      NSTO = LMAX - LFILE
      NINFF = MAX - MIN
      WRITE (3,201) LMAX,LFILE,NSTO,MAX,NINFF,M
201  FORMAT(/1X'CORE USE SUMMARY',27X,'MAXIMUM',17X,'USED',8X,'UNUSED'/
1 34X,'STORE (ELEMENT STORAGE)',14X,I10,1X,'(LMAX)',4X,I10,5X,I10/
2 34X,'INFF (ELEMENT DEFINITIONS)',10X,I10,1X,'(MAX)',5X,I10,5X,
3 I10)
      ERROR = .TRUE.
      RETURN
C
7    CONTINUE
C ZERO COMMON BLOCK/INSTR/,BUT DONT ZERO M
      DO 10 I=1,24
10   IC(I) = 0
      IC(21) = 4HSNGL
      NFSW = .FALSE.
      IF (.NOT. LDFLG) EMPTY = .FALSE.
      LDFLG = .FALSE.
      IF (EMPTY) GO TO 4
      EMPTY=.TRUE.
      READ(2,5) (ICARD(IW), IW=1,20)
5    FORMAT(20A4)
4    DECODE(4,3,ICARD)ISIGN
3    FORMAT(A1,3X)
C    IF ISIGN=P OR C, IT IS PAGE OR COMMENT AND NEEDS SPECIAL HANDLING
      IF (ISIGN .EQ. 1HP) GO TO 100
      IF (ISIGN .EQ. 1HC) GO TO 100
      IF (ISIGN .EQ. 1H.) GO TO 100
      DECODE(20,2,ICARD)ISIGN,NAME,OPNAME,KA,KB
2  FORMAT (A1,A4,2X,A4,1X,I3,1X,I3,1X)
C KA,KB BECOME -0 IF BLANK. RESET TO +0.
      IF (KA .EQ. 0) KA=0

```

```

      IF (KB .EQ. 0) KB=0
      OP = OPNUM(OPNAME)
      IF(OPNAME.EQ.3HRUN) M=M7END-ISAV7
      IF (ISIGN .EQ. 1H-) OP=-OP
      CALL WRTINF( IC,M )
      RETURN

```

```

100  OPNAME=4HNULL
      RETURN
      END

```

```

C ***** REF *****
      SUBROUTINE REF(M, N)
      IMPLICIT REAL*8 (A-H,O-Z)
C      COMPUTES MATRIX REFLECTION
C      REFLECT MATRIX AS 7 X 7, THEN STORE ACCORDING TO KIND
      INCLUDE 'DIM.CCC'
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION RX(2,3),RY(2,3),RW(3),T(49),TK(7,7),RPQ(4),RWT(3)
      DIMENSION ML(3)
      DATA TK /49*0.0/
C
C      COMPUTE REFLECTION BY MULTIPLYING MATRICES
C      K      M(INV)      K(INV)
C      WHERE K IS MATRIX TK AND K = K(INV)
C
C      GET INVERSE OF MATRIX
      NN = -N
      CALL RTRV7(NN, T, RW)
      CALL STOR7(M7END-2, T, RW)
C
C      FORM K MATRIX AND MULTIPLY
      TK(1,1) = 1.
      TK(2,2) = -1.
      TK(3,3) = 1.
      TK(4,4) = -1.
      TK(5,5) = -1.
      TK(6,6) = 1.
      TK(7,7) = 1.
      DO 1 I = 1,3
        RWT(I) = 0.
1      CONTINUE
      NT = M7END - 1
      CALL STOR7(NT, TK, RWT)
      ML(1) = NT
      ML(2) = M7END - 2
      ML(3) = NT
      CALL MMM(M, 3, ML)
C
C      LENGTH SHOULD BE SAME AS FIRST MATRIX, NOT THAT OF INVERSE.
C      REVERSE SIGN OF LENGTH
      RW(1) = -RW(1)
      RW(2) = -RW(2)
      RW(3) = -RW(3)

```

```

CALL STDAT(M, 5, 1, 3, RW)
RETURN
END

```

C ***** REPBCD *****

```

SUBROUTINE REPBCD(M,J,IB)
IMPLICIT REAL*8 (A-H,O-Z)

```

C REPLACES BCD DATA FROM B INTO INDEX J OF ELEMENT M BCD DATA

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
LOCI=INFF(16,M)*2-1
LOC=LOCI+J-1
ISTORE(LOC)=IB
RETURN
END

```

C ***** REPFLT *****

```

SUBROUTINE REPFLT(M,J,F)
IMPLICIT REAL*8 (A-H,O-Z)

```

C REPLACES FLOATING POINT DATA FROM F INTO INDEX J OF ELEMENT M

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
LOC=INFF(14,M)+J-1
STORE(LOC)=F
RETURN
END

```

C ***** REPINT *****

```

SUBROUTINE REPINT(M,J,I)
IMPLICIT REAL*8 (A-H,O-Z)

```

C REPLACES INTEGER DATA FROM I INTO INDEX J OF ELEMENT INTEGER DATA

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
LOCI=INFF(12,M)*2-1
LOC=LOCI+J-1
ISTORE( LOC )=I
RETURN
END

```

C ***** REPL *****

```

SUBROUTINE REPL(M)
IMPLICIT REAL*8 (A-H,O-Z)

```

C REPLACE A VALUE IN A PREVIOUSLY DEFINED INSTRUCTION

C THE 2ND BCD PARAMETER TELLS WHAT TYPE.

C F FOR FL.PT., H FOR HOLLERITH(BCD), I FOR INTEGER, K FOR KA OR KB

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'CONTRL.CCC'

```

```

INTEGER BDAT

```



```

      INTEGER doIBLANK/'      '/,doIF/'F'/,doIH/'H'/,doII/'I'/,doIKA/'KA'/
*      ,doIKB/'KB'/

      NM = BDAT(M,1)
      N = MDAT(M,1)
      KT = BDAT(M,2)
      K = INFF(4,M)
C      IF KT IS BLANK, ASSUME FLOATING POINT.
      IF (KT.EQ.doIBLANK) GO TO 100
      IF (KT.EQ.doIF) GO TO 100
      IF (KT.EQ.doIH) GO TO 110
      IF (KT.EQ.doII) GO TO 115
      IF (KT.EQ.doIKA) GO TO 95
      IF (KT.NE.doIKB) GO TO 120
      IK = 5
      GO TO 96
95      IK = 4
96      IP = IDAT(M,1)
      INFF(IK,N) = IP
      IF (MODE.EQ.3) WRITE(3,97) KT,NM,IP
97      FORMAT (6H *** ,8X,'REPLACE ',A3,' OF ',A5,' WITH ',I5)
      RETURN

C      FLOATING POINT
C      IF SYMBOLIC FLOATING POINT, ITS STRUCTURE IS MAINTAINED.
100      F = FLDAT(M,1)
      CALL REPFLT(N,K,F)
      IF (MODE.EQ.3) WRITE(3,101) K,NM,F
101      FORMAT (6H *** ,8X,'REPLACE ',I3,' OF ',A5,' WITH ',F10.6)
      RETURN

C      HOLLERITH (BCD)
110      IP = BDAT(M,3)
      CALL REPBCD(N,K,IP)
      IF (MODE.EQ.3) WRITE(3,111) K,NM,IP
111      FORMAT (6H *** ,8X,'REPLACE ',I3,' OF ',A5,' WITH ',A5)
      RETURN

C      INTEGER
115      IP = IDAT(M,1)
      CALL REPINT(N,K,IP)
      IF (MODE.EQ.3) WRITE(3,116) K,NM,IP
116      FORMAT (6H *** ,8X,'REPLACE ',I3,' OF ',A5,' WITH ',I5)
      RETURN

C      ERROR
120      WRITE(3,121) KT
121      FORMAT (//6H *** , 'ERROR IN REPL INSTRUCTION. ',A5,' IS AN UNKNO
1WN TYPE.')
130      ERROR = .TRUE.
      RETURN
      END
C ***** REPLQ *****
      SUBROUTINE REPLQ(M,L,J,F)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C PLACES F INTO INDEX J OF LQ STORAGE OF ELEMENT M
C L=1 FOR LQ      L=2 FOR LQ2    L=3 FOR LQ3
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  LM = 10
  IF (L.EQ.2) LM = 19
  IF (L.EQ.3) LM = 24
12 LOC = INFF(LM,M) + J - 1
C IF LQ3 OR LQ2 HAVE NOT BEEN SET, GET LQ2 OR LQ
  IF (LOC.NE.0) GO TO 15
  IF (LM.EQ.10) GO TO 15
  IF (LM.EQ.24) LM = 19
  IF (LM.EQ.19) LM = 10
  GO TO 12
15 STORE(LOC) = F
  RETURN
  END

C ***** RESRV *****
  SUBROUTINE RESRV(M0,KKND,IQ,N1,I1arg,N2arg,I2arg,N3arg,
    *I3arg,N4arg,I4arg,N5arg,I5arg,N6arg,I6arg,
    I1N7arg,I7arg,N8arg,I8arg,N9arg,I9arg,I10arg,I10arg)
  IMPLICIT REAL*8 (A-H,O-Z)
C MAKES RESERVATIONS FOR INPUT
C SEE COMMENTS IN LOAD FOR VARIABLE INPUT
C                                     THE NJ INTEGERS
C   NSETS = NUMBER OF SETS OF INPUT DATA   IF IJ= 0 SPECIFY NJ-S OF
C   NJ     = NUMBER OF DATA OF JTH SET      SUBSEQUENT SETS
C   IJ     = TYPE OF DATA OF JTH SET.       = 1 FLOATING F10.6
C                                           = 2 BCD      A5
C ONLY THE FIRST SET, J=1, MAY BE OF THE TYPE IJ=0. = 3 INTEGER 15
C SEE EXAMPLES IN LOAD.                       = 4 SKIP    10X
C
  INCLUDE 'BINFF.CCC'
  INCLUDE 'INSTR.CCC'
  INCLUDE 'STORE.CCC'
  INCLUDE 'CONTRL.CCC'
  INCLUDE 'DIM.CCC'
  LOGICAL ROWSW,COLSW,NQSW,CYSW
  LOGICAL*1 LGNVAR(60),MINUS/1H-/ ,NUM
  LOGICAL*1 ICVT(10)/'0','1','2','3','4','5','6','7','8','9'/
  DIMENSION ND(10),I(10),NVAR(10),NKIND(10),INF(24)
  EQUIVALENCE (LGNVAR(1),ICARD(6))
  DATA NKIND /3HMXV,3HPVC,3HMAT,2HM6,2HR7,3HROT,3HMAP,
13HPRD,4HSCAL,1H /
  INTEGER do1ASTRK/4H* / ,NEGZER/--32767/
  INTEGER do1BLNK/4H /
  INTEGER do1FXPT/4HFXPT/,do1SNGL/4HSNGL/,do1CYB/3HCYB/,do1CYC/3HCYC
  * /
  INTEGER do1CYA /'CYA '/, do1BEST /'BEST'/
  INTEGER do1TAB/3HTAB/

C
  I1=I1arg
  N2=N2arg
  I2=I2arg

```

```

N3=N3arg
I3=I3arg
N4=N4arg
I4=I4arg
N5=N5arg
I5=I5arg
N6=N6arg
I6=I6arg
N7=N7arg
I7=I7arg
N8=N8arg
I8=I8arg
N9=N9arg
I9=I9arg
N10=N10arg
I10=I10arg

```

```
GO TO 2
```

C

```
ENTRY RESRVA(M0,KKND,IQ,N1,I1arg,N2arg,I2arg,N3arg,I3arg,N4arg)
```

```
I1=I1arg
```

```
N2=N2arg
```

```
I2=I2arg
```

```
N3=N3arg
```

```
I3=I3arg
```

```
N4=N4arg
```

```
IF (N4 .EQ. 0) GO TO 2
```

```
WRITE(3, 1009) N4, M0
```

```
1009 FORMAT(' RESRVA ERROR', 2I6)
```

```
GO TO 1
```

C

```
ENTRY RESRV8(M0,KKND,IQ,N1,I1arg,N2arg,I2arg,N3arg)
```

```
I1=I1arg
```

```
N2=N2arg
```

```
I2=I2arg
```

```
N3=N3arg
```

```
IF (N3 .EQ. 0) GO TO 2
```

```
WRITE(3, 1010) N3, M0
```

```
1010 FORMAT(' RESRV8 ERROR', 2I6)
```

```
GO TO 1
```

C

```
ENTRY RESRV6(M0,KKND,IQ,N1,I1arg,N2arg)
```

```
I1=I1arg
```

```
N2=N2arg
```

```
IF (N2 .EQ. 0) GO TO 2
```

```
WRITE(3, 1011) N2, M0
```

```
1011 FORMAT(' RESRV6 ERROR', 2I6)
```

```
GO TO 1
```

C

```
ENTRY RESRV4(M0,KKND,IQ,N1)
```

```
IF (N1 .EQ. 0) GO TO 2
```

```
WRITE(3, 1012) N1, M0
```

```
1012 FORMAT(' RESRV4 ERROR', 2I6)
```

C

```
1 CONTINUE
```

```
WRITE(3, 1013) M0, KKND, IQ
```

```

WRITE(3, 1013) N1,N2,N3,N4,N5,N6,N7,N8,N9,N10
WRITE(3, 1013) I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
1013 FORMAT(' ', 10I6)
CALL ERRTRA
CALL EXIT
C
2 CONTINUE
C
IF(MODE.EQ.3) GO TO 71
IF (.NOT.RSRV) GO TO 71
NQSW=.FALSE.
ROWSW=.FALSE.
COLSW=.FALSE.
IF (ROWS.NE.0) ROWSW=.TRUE.
IF (COLS.NE.0) COLSW=.TRUE.
C STORE DATA SPECIFICATION NUMBERS IN ARRAYS ND(J) AND I(J)
CALL ARGINP(N1,I1,N2,I2,N3,I3,N4,I4,N5,I5,N6,I6,N7,I7,N8,I8,N9,I9,
1N10,I10,ND,I,NSETS)
8 NIN =0
NFL =0
NBC =0
LQ2 = LQ
LQ3 = LQ
LQ = LFILE+ 1
C IF IQ.GE.0, NQ=IQ=NO. LOCATIONS RESERVED STARTING AT LQ
C IF IQ.EQ.-0, NQ WILL BE CALCULATED ACCORDING TO KIND
C IF IQ.LE.-1, DABS(IQ) REFERS TO THE IQ-TH SET OF DATA.
C LQ WILL BE SET = LIN=LBC,OR LFL, DEPENDING ON DATA TYPE
IF(IQ.EQ.0.OR.IQ.EQ.NEGZER) GO TO 10
IF (IQ.GE.0) GO TO 15
C IQ IS NEGATIVE NUMBER.
C NQ WILL BE DETERMINED BY THE IQ-TH SET OF INPUT DATA
NQ = 0
GO TO 16
C DETERMINE SIGN OF ZERO
10 CONTINUE
IF (IQ .EQ. NEGZER) NQSW = .TRUE.
IF (NQSW) GO TO 18
C IQ IS POSITIVE NUMBER. SET NQ=IQ
15 NQ = IQ
C IF KIND NOT SPECIFIED,SET IT TO OPNAME
16 IF (KKND.EQ.0) KIND = INFF(1,M0)
IF (KIND.EQ.0) KIND = NKIND(KKND)
IF (OPNAME.NE.doIFXPT) GO TO 30
C FXPT. SET UP 3 LQ LOCATIONS
LQ2 = LQ + KADD +49
LQ3 = LQ2 + 35
NQ = LQ3 + 15 - LQ
GO TO 30
C IQ IS -0. NQ DETERMINED BY KIND
18 IF(KKND.NE.doIBLNK) GO TO 181
C KIND IS UNKNOWN. SET NQ = 0. LOAD WILL SET LQ AND NQ AFTER LOADING.
KIND = KKND
NQ = 0
GO TO 30

```

C KIND IS KNOWN. FIND OUT WHICH ONE.

```

181  NK = 1
      DO 19 J=1,10
        IF (KKND.EQ.NK) GO TO 20
        NK=NK+1
19   CONTINUE
      PRINT 110
110  FORMAT (/5X,25H ILLEGAL KIND ENCOUNTERED//)
      ERROR=.TRUE.
      GO TO 71
20   KIND=NKIND(NK)
      GO TO (21,22,23,24,25,26,25,25,29),NK

```

C KIND IS MXPMY. MATRIX PAIRS RX AND RY

```

21   NQ = 12 + KADDR
      IF (.NOT.ROWSW) ROWS=2
      IF (.NOT.COLSW) COLS=7
      GO TO 30

```

C KIND IS PVECS, PARTICLE VECTOR

C IF NOT OTHERWISE INDICATED, 7 ROWS BY KA COLUMNS

```

22   NQ = 7*KA
      IF (.NOT.ROWSW) ROWS = 7
      IF (.NOT.COLSW) COLS= KA
      GO TO 30

```

C KIND IS MAP OR PRD. SAVE SPACE

C KIND IS SCALAR

```

29   NQ = 1
      IF (TYPE.NE.DOISNGL) NQ = 2
      GO TO 30

```

C KIND IS MAT, MATRIX OK H ROWS AND L COLUMNS

```

23   NQ = KADD
      GO TO 28

```

C KIND IS 6X6 MATRIX

```

24   NQ = 22 + KADD
28   IF(.NOT.ROWSW) ROWS = KA
      IF (.NOT.COLSW) COLS=KB
      GO TO 30

```

C KIND IS R7, 7X7 MATRIX

```

25   NQ = 49 + KADD
      ROWS = 7
      COLS = 7
      GO TO 30

```

C KIND IS ROT,ROTATION

```

26   NQ = KADD
      KIND = NKIND(NK)

```

C SET LFMX FOR NSETS=0 CASE. IF NSETS NOT 0, LFMX COMPUTED BELOW.

```

30   CONTINUE
1004 FORMAT(' ', 10I6)
      LFMX=LQ+NQ-1
      IF(NSETS.EQ.0) GO TO 60
      IF (I(NSETS).GE.0) GO TO 31

```

C NUMBER OF DATA IN LAST SET IS INDEFINITE.

```

      INDEF=.TRUE.
      I(NSETS) = -I(NSETS)

```

```

31   IF(I(1).NE.0) GO TO 32

```

C STORE VARIABLE-INPUT-LENGTH NUMBERS IN NVAR(I).

```

        DECODE(60,100,ICARD(6))(NVAR(K),K=1,N1)
100  FORMAT(12I5)
C CALCULATE NUMBER OF DATA OF EACH TYPE.
32  DO 44 J=1,NSETS
      NAB = IABS(ND(J))
      IF (ND(J).LT.0) ND(J) = NVAR(NAB)
      IJ = I(J) + 1
      GO TO (44,41,42,43,44),IJ
41  NFL=NFL + ND(J)
      GO TO 44
42  NBC=NBC + ND(J)
      GO TO 44
43  NIN=NIN + ND(J)
44  CONTINUE
C    IF A CYX INSTRUCTION, SET UP LQ2
      CYSW=.FALSE.
      IF ( (OPNAME .EQ. doICYB) .OR. (OPNAME .EQ. doICYC) ) CYSW=.TRUE.
      *
      IF ( (OPNAME .EQ. doICYA) .OR. (OPNAME .EQ. doIBEST) ) CYSW=.TRUE
      *
      IF (.NOT. CYSW) GO TO 45
      LQ2=LQ+NQ
      NQ = NQ + 10
45  CONTINUE
      IF (OPNAME .NE. doITAB) GO TO 46
C    SET UP EXTRA LQ FOR TABLE USE
      LQ2=LQ+NQ
      LQ3=LQ2+10
      NQ=NQ+20
46  CONTINUE
C CALCULATE FILE LOCATIONS.
C ORDER SETS WITH FLOATING POINT AFTER LQ
      NIWS =(NIN+1)/2
      NBWS =(NBC+1)/2
53  LFL = LQ + NQ
      LBC = LFL + NFL
      LIN = LBC + NBWS
      LFMX = LIN + NIWS - 1
60  CONTINUE
      NTOT = LFMX - LFILE
      IF (IQ.GE.0 .OR. IQ .EQ. NEGZER) GO TO 65
C RESET NQ. RESET LQ ACCORDING TO TYPE OF DATA IN IQ-TH SET
C IQ REFERS TO THE ABS IQ-TH SET OF INPUT DATA
      IQ=-IQ
      NQ=ND(IQ)
      IN=I(IQ)
      GO TO (61,62,63),IN
61  LQ=LFL
      NQ = NFL
      GO TO 65
62  LQ=LBC
      GO TO 65
63  LQ=LIN
C PACK LOCATIONS AND LENGTHS INTO INFO(I,N).
65  IF(NIN.EQ.0) LIN=0

```

```

      IF(NFL.EQ.0) LFL=0
      IF(NBC.EQ.0) LBC=0
      DO 70 J=1,9
70  INF(J) = doIASTRK
      INF(10) = LQ
      INF(11) = NQ
      INF(12) = LIN
      INF(13) = NIN
      INF(14) = LFL
      INF(15) = NFL
      INF(16) = LBC
      INF(17) = NBC
      INF(18) = NTOT
      INF(19)=LQ2
      INF(20) = KIND
      INF(21)=TYPE
      INF(22) = ROWS
      INF(23) = COLS
      INF(24) = LQ3
      IF (M0.GT.0) CALL WRTINF(INF,M0)
C  RESET BOTTOM OF FILE,LFILE,TO BOTTOM OF MTH SECTION OF THE FILE.
      LFILE= LFMX
C  CHECK FOR STORAGE OVERFLOW
      NSTO = LMAX - LFILE
      NINFF = MAX - MIN
      IF (LFILE.LE.LMAX) GO TO 80
      WRITE (3,201)
      WRITE (3,202) LMAX,LFILE,NSTO,MAX,NINFF,M0
      ERROR = .TRUE.
80  IF (M0.GT.0) RETURN
      WRITE (3,200)
      IF (.NOT.ERROR) WRITE (3,202) LMAX,LFILE,NSTO,MAX,NINFF,M0
      ERROR = .TRUE.
200  FORMAT (6H0*****,'ELEMENTS OVERFLOWED. INCREASE INFF ARRAY AND '
1  'MAX.')
201  FORMAT (6H0*****,'STORE OVERFLOWED. INCREASE STORE ARRAY AND LMAX'
1  )
202  FORMAT(/1X'CORE USE SUMMARY',27X,'MAXIMUM',17X,'USED',8X,'UNUSED'/
1  34X,'STORE (ELEMENT STORAGE)',14X,I10,1X,'(LMAX)',4X,I10,5X,I10/
2  34X,'INFF (ELEMENT DEFINITIONS)',10X,I10,1X,'(MAX)',5X,I10,5X,
3  I10)
71  RETURN
      END

C ***** RESTOR *****
      SUBROUTINE RESTOR(K)
      IMPLICIT REAL*8 (A-H,O-Z)
CC  RESTORES A FIXED PARAMETER TO VARIABLE STATUS
CC  BY INSERTING IT INTO THE INTERNAL PARAMETER LIST AT THE
CC  APPROXIMATE PLACE.
CC
CCALL COMMONT
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET

```

```

COMMON /UNIT / ISYSRD      ,ISYSWR      ,ISYSPU
COMMON/FIX/XS(15),XTS(15),WTS(15),NPFIX,IPFIX(15)
COMMON /TITLE / TITLE(13)  ,DATE(2)    ,ISW(7)      ,NBLOCK
COMMON /MINIMA/ AMIN      ,UP          ,NEWMIN      ,ITAU      ,SIGMA
C—      K = 0 MEANS RESTORE ALL PARAMETERS
C—      K = 1 MEANS RESTORE THE LAST PARAMETER FIXED
C—      K = -1 MEANS RESTORE EXTERNAL PARAMETER I (IF POSSIBLE)
C—      IQ = FIX-LOCATION WHERE INTERNAL PARAMETERS WERE STORED
C—      IR = EXTERNAL NUMBER OF PARAMETER BEING RESTORED
C—      IS = INTERNAL NUMBER OF PARAMETER BEING RESTORED
      IF (K .GT. 1) WRITE (ISYSWR,510)
      IF (NPFIX .LT. 1) WRITE (ISYSWR,500)
      IF (K.EQ.1 .OR. K.EQ.0) GO TO 50
C      RELEASE PARAMETER WITH SPECIFIED EXTERNAL NUMBER
      KA = IABS(K)
      IF (LCORSP(KA) .EQ. 0) GO TO 15
      WRITE (ISYSWR,540)
540 FORMAT (49H0ERROR. PARAMETER SPECIFIED IS ALREADY VARIABLE. /)
      RETURN
      15 IF (NPFIX .LT. 1) GO TO 21
      DO 20 IK= 1, NPFIX
      IF (IPFIX(IK) .EQ. KA) GO TO 24
      20 CONTINUE
      21 WRITE (ISYSWR,530)
530 FORMAT (53H0ERROR. PARAMETER SPECIFIED HAS NEVER BEEN VARIABLE./)
      RETURN
      24 IF (IK .EQ. NPFIX) GO TO 50
C      MOVE SPECIFIED PARAMETER TO END OF LIST
      IPSAV = IPFIX(IK)
      XSSAV = XS(IK)
      XTSSAV = XTS(IK)
      WTSSAV = WTS(IK)
      IKP1 = IK + 1
      DO 30 I= IKP1,NPFIX
      IPFIX(I-1) = IPFIX(I)
      XS(I-1) = XS(I)
      XTS(I-1) = XTS(I)
      30 WTS(I-1) = WTS(I)
      IPFIX(NPFIX) = IPSAV
      XS(NPFIX) = XSSAV
      XTS(NPFIX) = XTSSAV
      WTS(NPFIX) = WTSSAV
C      RESTORE LAST PARAMETER IN LIST — IPFIX(NPFIX)
      50 CONTINUE
      IF (NPFIX .LT. 1) GO TO 300
      IR = IPFIX(NPFIX)
      IS = 0
      DO 100 IJ= IR, NU
      IK = NU + IR - IJ
      IF (LCORSP(IK)) 100,100,85
      85 LC = LCORSP(IK) + 1
      IS = LC - 1
      LCORSP(IK) = LC
      X(LC) = X(LC-1)
      XT(LC) = XT(LC-1)

```



```

      DIRIN(LC) = DIRIN(LC-1)
100 CONTINUE
      NPAR = NPAR + 1
      IF (IS .EQ. 0) IS = NPAR
      LCORSP(IR) = IS
      IQ = NPFIX
      X(IS) = XS(IQ)
      XT(IS) = XTS(IQ)
      DIRIN(IS) = WTS(IQ)
      NPFIX = NPFIX - 1
      ISW(2) = 0
      IF (ITAU .LT. 1) WRITE(ISYSWR,520) IR,NAM(IR)
      IF (K.EQ.0) GO TO 50
300 RETURN
500 FORMAT ( 61H0ERROR IN CALL TO RESTOR.  THERE ARE NO MORE FIXED PA
      1RAMETERS/)
510 FORMAT ( 52H0ERROR IN CALL TO RESTOR.  ARGUMENT GREATER THAN ONE/
      1)
520 FORMAT (20X, 9HPARAMETER,I4,2H, ,A10, 22H RESTORED TO VARIABLE.
      1)
      END
C ***** REVMAT *****

```

```

      SUBROUTINE REVMAT(N,M,V)
      IMPLICIT REAL*8 (A-H,O-Z)

C  MAPS PARTICLE VECTOR V BY ELEMENT WITH INDEX M.
C  ELEMENT MAY BE ANY LINEAR ELEMENT OR SEXTUPOLE.
C  IF THE ELEMENT IS NOT COMPOSITE (E.G. AN MMM), LINEARIZED MATRICES
C  THAT GIVE THE TRANSFORMATION RELATIVE TO V ARE CALCULATED AND
C  STORED AT INDEX N.
C  IF M IS A COMPOSITE ELEMENT, A NEW MATRIX IS NOT CALCULATED,
C  THE OLD ONE IS STORED AT N.

C  ELEMENTS DEFINED BY MAG AND SXTIP WILL BE RECALCULATED BY REVMAT,
C  INCLUDING BR=BR*(1+DP/P).

C  **NOTE**  THIS ROUTINE USES MEND-3, MEND-4, AND MEND-5 (3X3 LOCS)

```

```

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      INCLUDE 'DIM.CCC'
      INCLUDE 'CORB.CCC'
      COMMON/MN/ML(3)
      DIMENSION PAR(6),T(7,7),RW(3),V(1),V0(7),W(7),U(7)
      EQUIVALENCE (ANG1,IANG1),(ANG2,IANG2),(BR,PAR(3))
      INCLUDE 'CONST.CCC'

      NOP = INFF(1,M)
C  MOVE
      IF (NOP.NE.4HMOVE) GO TO 100
      CALL MOVE(M,N,V)

```

RETURN

C

```
100 DO 2 I=1,7
    U(I)=V(I)
2    V0(I)=V(I)
```

C SEXTUPOLE

```
IF (NOP.NE.4HSXTP) GO TO 1
CALL DATA(M,1,1,4,PAR)
INFF(4,N) = INFF(4,M)
INFF(5,N) = INFF(5,M)
CALL SXTP(N,V,PAR)
RETURN
```

C MAGNET DEFINED BY MAG INSTRUCTION

```
1 IF (NOP.NE.3HMAG) GO TO 7
CALL DATA(M,1,1,6,PAR)
GRAD=PAR(2)
BZ=PAR(4)
IRECT=0
ANG1=PAR(5)
ANG2=PAR(6)
IF ( (IANG1.EQ.1H$).OR.(IANG2.EQ.1H$) ) IRECT=1
IF ( (GRAD.NE.0.).OR.(IRECT.EQ.0.).OR.(BZ.EQ.0.) ) GO TO 11
```

C RECTANGULAR,ZERO-GRADIENT BENDING MAGNET

```
CALL BEND(N,V,PAR,2)
GO TO 9
```

```
11 CONTINUE
IF (BZ.NE.0.) GO TO 111
```

C QUADRUPOLE

```
CALL MODQ(M,N,V,W)
GO TO 45
111 BR=BR*(1.D0+V(6))
CALL MAGNET(N,PAR)
CALL RTRV7(N,T,RW)
CALL MXV7(T,V,W)
```

C BENDING MAGNET

```
THETA = RW(2)
IF (IRECT.EQ.1) GO TO 3
E1=PAR(5)*RADEG
E2=PAR(6)*RADEG
GO TO 4
3 E1=.5D0*THETA
E2=E1
4 E1T=E1+V(2)
E2T=E2-W(2)
X1=V(1)
X2=W(1)
IF (BZ.GE.0.) GO TO 40
E1T=E1-V(2)
E2T=E2+W(2)
X1=-X1
X2=-X2
```

```

40  PAR(5)=E1T/RADEG
    PAR(6)=E2T/RADEG
    D1=X1*DSIN(E1)/DCOS(E1T)
    D2=X2*DSIN(E2)/DCOS(E2T)
    DL=V(5)-W(5)
    PAR(1) = PAR(1) + DL - D1 - D2
    THETA = THETA + V(2) - W(2)
    RHO = PAR(1)/THETA
    PAR(4) = BR/RHO
    DO 5 I=1,3
5    ML(I) = MEND - 2 - I
    CALL DRIFT(MEND-3,D1)
    CALL MAGNET(MEND-4,PAR)
    CALL DRIFT(MEND-5,D2)
    CALL MM(N,3,ML)
45  DO 50 I=1,5
50  V(I) = W(I)
    GO TO 9

```

C DRIFT

```

7    IF(NOP.NE.3HDRF) GO TO 8
    EL = FDAT(M,1)
    EL=EL*DSQRT(1.D0+V(2)*V(2))
    CALL DRIFT(N,EL)
    GO TO 10

```

C KICK

```

8    IF (NOP.NE.4HKICK) GO TO 82
    NAKICK = INFF(2,M)
    IF (NAKICK.NE.NACORH) GO TO 81
    TKH = TH(IIH)
    CALL STDAT(M,1,3,1,TKH)
    CALL KICK(M)
    IIH = IIH + 1
    IF (IIH.GT.NNH) IIH = 1
    GO TO 10
81   IF (NAKICK.NE.NACORV) GO TO 10
    TKV = TV(IIV)
    CALL STDAT(M,1,3,1,TKV)
    CALL KICK(M)
    IIV = IIV + 1
    IF (IIV.GT.NNV) IIV=1
    GO TO 10

```

C OTHER LINEAR ELEMENTS—~~MM~~, ETC.

```

82   CONTINUE
10   CALL RTRV7(M,T,RW)
    CALL MXV7(T,V,V)
    CALL STOR7(N,T,RW)
    RETURN

```

```

9    CALL RTRV7(N,T,RW)
91   CALL MXV7(T,V0,W)
    DO 12 I=1,5
12   T(I,7)=V(I)-W(I)
    CALL STOR7(N,T,RW)

```

```

RETURN
END

```

```

C ***** ROTATE *****
      SUBROUTINE ROTATE(THETA,T,TR,RW)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C          ROTATE MATRIX T BY THETA RADIANS AND FORM NEW MATRIX TR.

```

```

      INCLUDE 'DIM.CCC'
      INCLUDE 'BMI.CCC'

```

```

      DIMENSION T(1),TR(1),RW(1),RW1(3)
      DATA RW1/3*0.0/

```

```

C          COMPUTE ROTATION MATRIX R+ AND R-

```

```

      THETM = -THETA
      CALL ROTZ(THETA,TR)
      M1 = M7END - 1
      CALL STOR7(M1,TR,RW1)
      MI(1) = M1
      M1 = M1 - 1
      CALL STOR7(M1,T,RW)
      MI(2) = M1
      M1 = M1 - 1
      CALL ROTZ(THETM,TR)
      CALL STOR7(M1,TR,RW1)
      MI(3) = M1
      M1 = M7END
      CALL MMM(M1,3,MI)
      CALL RTRV7(M1,TR,RW)
      RETURN
      END

```

```

C ***** ROTM *****
      SUBROUTINE ROTM(M,MB,THETA)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C          ROTATE MATRIX MB BY THETA RADIANS AND STORE IN MATRIX M.

```

```

C          MB AND M ARE INDECESES OF MATRICES

```

```

      DIMENSION T(49),TR(49),RW(3)

```

```

      CALL RTRV7(MB,T,RW)
      CALL ROTATE (THETA,T,TR,RW)
      CALL STOR7(M,TR,RW)
      RETURN
      END

```

```

C ***** ROTZ *****
      SUBROUTINE ROTZ (THET,T)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C          DEFINE A MATRIX REPRESENTING A ROTATION

```

```

C          ALONG THE Z AXIS. THET IS IN RADIANS.

```

```

      DIMENSION T(7,7)
      DO 10 I=1,7
      DO 10 J=1,7

```

```

      T(I,J)=0.
10  CONTINUE
      SN=DSIN(THET)
      CS=DCOS(THET)
      T(1,1)=CS
      T(2,2)=CS
      T(3,3)=CS
      T(4,4)=CS
      T(1,3)=SN
      T(2,4) = SN
      T(3,1) = -SN
      T(4,2) = -SN
      T(5,5) = 1.
      T(6,6) = 1.
      T(7,7) =1.
      RETURN
      END

C*****ROW5*****
      SUBROUTINE ROW5(T)
      IMPLICIT REAL*8 (A-H,O-Z)
C  COMPUTES ROW 5 OF 7X7 MATRIX T

      DIMENSION T(7,1)
      DO 10 I=1,4
10  T(5,I)=T(2,I)*T(1,6)-T(1,I)*T(2,6)+T(4,I)*T(3,6)-T(3,I)*T(4,6)
      RETURN
      END

C ***** RPBL *****
      SUBROUTINE RPBL(M)
C  REPLACE BEAM LINE
C  CREATES A NEW BEAM LINE FROM AN OLD ONE, REPLACING SPECIFIED
C  ELEMENTS WITH OTHERS.
C .C      BML      QF  OO  QD  QD  QD  OO  QF  ...
C      RPBL      .C  .C1 QF  QF1 QD  QD1 ....
C
C  MAKES A NEW BEAM LINE NAMED .C1 LIKE .C, BUT WITH QF REPLACED BY
C  QF1, QD REPLACED BY QD1, ETC.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      INCLUDE 'STORE.CCC'

      DIMENSION NPAR(300),NP1(150),NP2(150),IC(24)
      INTEGER ELNUM,OLDBL,OPNUM,OPNAME,OP,BDAT
      EQUIVALENCE (OPNAME,IC(1)),(NAME,IC(2)),(OP,IC(3))
      EQUIVALENCE (KA,IC(4)),(KB,IC(5))
      DATA IBLNK/1H /

C  RETRIEVE DATA.  MAKE PARALLEL ARRAYS, NP1,NP2 FOR REPLACING.
      OLDBL = BDAT(M,1)
      NEWBL = BDAT(M,2)
      NBDAT = INFF(17,M) -2
      CALL DATA (M,2,3,NBDAT,NPAR)
      N2 = NBDAT/2
      DO 10 I=1,N2,2

```

```

      NP1(I) = NPAR(I)
10    NP2(I) = NPAR(I+1)
C    RETRIEVE OLD BEAM LINE
      NB1 = ELNUM(OLDBL)
      NI = INFF(17,NB1)
      CALL DATA (NB1,2,1,NI,NPAR)
C    SEARCH NPAR FOR NAMES IN NP1.  WHEN FOUND, REPLACE WITH NP2.
      DO 20 I=1,NI
        IF (NPAR(I).EQ.IBLNK) GO TO 20
      DO 15 IB=1,N2
        IF (NP1(IB).EQ.IBLNK) GO TO 15
        IF (NPAR(I).EQ.NP1(IB)) NPAR(I) = NP2(IB)
15    CONTINUE
20    CONTINUE
C
C    MAKE UP NEW INFF AND STORE NEW BEAM LINE
      MIN = MIN - 1
      CALL INFW(MIN,8,M)
      M1 = MIN
      DO 25 I=1,24
25    IC(I) = 0
      IC(21) = 4HSNGL
      OPNAME = 3HBML
      NAME = NEWBL
      OP = OPNUM(IC(1))
      KA = 0
      KB = 0
      NI2 = NI*2
      CALL WRTINF(IC,M1)
      CALL RESRV6(M1,0,0,NI2,2,0)
      CALL STDAT(M1,2,1,NI,NPAR)
C    RESET INFF
      INFF(17,M1) = INFF(18,M1)
      INFF(19,M1) = 0
      INFF(24,M1) = 0
C    WRITE CARD IMAGE
      NPR = 12
      IF (NPR.GT.NI) NPR = NI
      WRITE (3,106) NAME,OPNAME,KA,KB,(NPAR(I),I=1,NPR)
106  FORMAT (6H  *** ,1X,A5,2X,A5,1X,I3,1X,I3,5H // ,12(A4,1X))
50   IF (NPR.GE.NI) GO TO 120
      II = NPR + 1
      NPR = NPR + 12
      IF (NPR.GT.NI) NPR=NI
      WRITE (3,107) (NPAR(I),I=II,NPR)
107  FORMAT (6H  * ,21X,5H // ,12(A4,1X))
      GO TO 50
C    READJUST M
120  M = M1
      RETURN
      END

C ***** * RSVMAT *****
      SUBROUTINE RSVMAT
      IMPLICIT REAL*8 (A-Z)

```

```

C      RESERVES SPACE FOR KA MATRICES, BEGINNING WITH INDEX M - 1
      INCLUDE 'INSTR.CCC'
      INCLUDE 'CONTRL.CCC'

      DIMENSION LIST(25)
      INTEGER doIDRF/'DRF'/',doIMAG/'MAG'/'

      IG = 0
      IF (OPNAME.EQ.doIDRF) IG = 1
      IF (OPNAME.EQ.doIMAG) IG = 2
      IF (IG.LT.1) GO TO 25

C      GENERATE NUMBERED NAMES
      CALL GENNUM(NAME,KA,LIST)
C      RESERVE ACCORDING TO OPNAME
      M1 = M - 1
      K1 = 0
      K2 = 0
      DO 20 I=1,KA
      GO TO (10,15), IG
10     CALL DRFRSV(M1,LIST(I),K1,K2,0)
      GO TO 18
15     CALL MAGRSV(M1,LIST(I),K1,K2,0)
18     M1 = M1 - 1
20     CONTINUE
      RETURN

25     ERROR = .TRUE.
      WRITE (3,1) OPNAME
1     FORMAT (5X,3H***,1X,'ERROR',1X,A5,1X,'IS NOT A DRF OR MAG.']/
1     9X,'ROUTINE RSVMAT CANNOT RESERVE MATRIX SPACE FOR IT.')]
      RETURN
      END

C ***** RTRV7 *****
      SUBROUTINE RTRV7(M,TT,RW)
      IMPLICIT REAL*8(A-H,O-Z)
C  RETRIEVES 7X7 MATRIX ELEMENTS
C  IF ELEMENT NUMBER IS -, RETURN INVERSE
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'DIM.CCC'

      LOGICAL INV
      DIMENSION TT(1)
      DIMENSION TM(49),RW(3),RX(2,3),RY(2,3),T(7,7),RPQ(4)
      EQUIVALENCE (T(1,1),TM(1))
      INTEGER doIROT/'ROT'/', doIMXY/'MXY'/'
      INCLUDE 'CONST.CCC'

      INV = .FALSE.
      DO 4 I=1,3
4     RW(I) = 0.
      IF (M)5,6,6
5     M = - M

```

```

      INV = .TRUE.
6     KIND = INFF(20,M)
      CALL DATA(M,5,1,3,RW)
C    IS KIND R7 OR ROT
      IF (KIND.EQ. doIROT) GO TO 40
      IF (KIND .NE. doIMXY) GO TO 28
C    KIND IS MXY. RETURN 2X3 MATRIX
      CALL RXY(M,RX,RY,RW)
      CALL DATA(M,5,4,4,RPQ)
      KD = 7
      CALL CNVMAT(KD,T,RX,RY,RW,RPQ)
      GO TO 50
C    KIND IS R7
28    LQ = INFF(10,M)
      NQ = INFF(11,M)
C    CHECK STORAGE
      NNQ = 49 + KADD
      IF (NNQ.GT.NQ) GO TO 45
      NJ = LQ + KADD - 1
      DO 35 I=1,49
      TM(I) = STORE(NJ + I)
35    CONTINUE
      GO TO 50
45    MN = INFF(2,M)
      WRITE(3,1) MN, NNQ, NQ, M
1     FORMAT (61H *** ERROR *** RETRIEVAL REQUEST EXCEEDS STORAGE RESERV
1E FOR A6, 3I6)
      ERROR = .TRUE.
      GO TO 57
C    KIND IS ROT
40    CONTINUE
      CALL DATA(M,1,1,1,THETA)
      THETA = THETA * RADEG
      DO 41 I=1,49
      TM(I) = 0.
41    CONTINUE
      CALL ROTZ(THETA,T)
50    IF (.NOT.INV) GO TO 52
      CALL TINV7(TT,TM)
      RW(1) = - RW(1)
      RW(2)=-RW(2)
      GO TO 56
52    DO 55 I=1,49
      TT(I) = TM(I)
55    CONTINUE
56    RW(3)=TT(40)
57    CONTINUE
      RETURN
      END
C ***** RXRY *****
      SUBROUTINE RXRY(M,DAT)
      IMPLICIT REAL*8 (A-H,O-Z)
C    READS MATRIX RX AND RY,DIMENSIONED (2,3) FROM CARDS.
C    STORES RX AND RY IN LQ(M)
      DIMENSION RX(2,3),RY(2,3),DAT(12),RW(3)

```



```

DATA RW /3*0./
C SETS UP MATRICES WHOSE ELEMENTS ARE READ FROM CARDS
  IX=1
  IY=7
  DO 1 I=1,2
  DO 1 J=1,3
  RX(I,J)=DAT(IX)
  RY(I,J)=DAT(IY)
  IX=IX+1
  IY=IY+1
1  CONTINUE
  CALL STXY(M,RX,RY,RW)
  RETURN
  END
C ***** RXY *****
  SUBROUTINE RXY(M,RX,RY,RW)
  IMPLICIT REAL*8 (A-H,O-Z)
C
C RETRIEVES X AND Y MATRICES FROM LQ(M). DIMENSIONS ARE (2,3) FOR EACH.
C RETURN RX AND RY
C IF ELEMENT NUMBER IS -, RETURN INVERSE
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'DIM.CCC'
  INCLUDE 'SWTCH.CCC'
  INCLUDE 'CONTRL.CCC'

  LOGICAL INV
  DIMENSION RX(2,3),RY(2,3),AX(2,3),AY(2,3),RW(3)
  DIMENSION TT(49),T(7,7),RPQ(4)
  EQUIVALENCE (T(1,1),TT(1))
C
  INV = .FALSE.
  IF (M.GE.0) GO TO 5
4  M = - M
  INV = .TRUE.
5  LQ = INFF(10,M)
  NQ = INFF(11,M)
  DO 7 J=1,3
  DO 3 I=1,2
  RX(I,J) = 0.
  RY(I,J) = 0.
3  CONTINUE
  RW(J) = 0.
7  CONTINUE
  KIND = INFF(20,M)
  IF(KIND.EQ.3HROT) GO TO 12
  IF(KIND.EQ.3HMX) GO TO 6
C KIND IS R7. RETRIEVE AND CONVERT
  NN = M
  IF (INV) NN = -NN
  CALL RTRV7 (NN,T,RW)
  KD = 3
  CALL CNVMAT(KD, T, RX, RY, RW, RPQ)
  RETURN

```

C KIND IS ROT

```

12  THETA = FDAT(M,1)
    CS = DCOS(THETA)
    DO 14 I=1,2
      RX(I,1) = CS
      RY(I,1) = CS
14  CONTINUE
    GO TO 24

```

C CHECK STORAGE

```

6   NNQ = 12+KADDR
    IF (NNQ .GT. NQ) GO TO 20
    NJ = LQ + KADDR
    NI = NJ + 6
    DO 10 IJ=1,3
      DO 10 II=1,2
        RX(II,IJ)=STORE(NJ)
        RY(II,IJ)=STORE(NI)
        NJ=NJ+1
        NI=NI+1
10  CONTINUE
    CALL DATA(M,5,1,3,RW)
24  IF (.NOT. INV) RETURN
    INV = .FALSE.
    DO 25 J=1,3
      DO 25 I=1,2
        AX(I,J) = RX(I,J)
25  AY(I,J) = RY(I,J)
    RX(1,1) = AX(2,2)
    RX(2,2) = AX(1,1)
    RX(1,2) = -AX(1,2)
    RX(2,1) = -AX(2,1)
    RX(1,3) = AX(1,2)*AX(2,3) - AX(2,2)*AX(1,3)
    RX(2,3) = AX(2,1)*AX(1,3) - AX(1,1)*AX(2,3)
    RY(1,1) = AY(2,2)
    RY(2,2) = AY(1,1)
    RY(1,2) = -AY(1,2)
    RY(2,1) = -AY(2,1)
    RY(1,3) = AY(1,2)*AY(2,3) - AY(2,2)*AY(1,3)
    RY(2,3) = AY(2,1)*AY(1,3) - AY(1,1)*AY(2,3)
    RW(1) = -RW(1)
    RW(2) = -RW(2)
    IF (BEND.NE.0.) RW(3) = - RW(3)
    RETURN

```

C

```

20  WRITE(3,1) M
1   FORMAT (66H *** ERROR *** RETRIEVAL REQUEST EXCEEDS STORAGE RESERV
1E AT INDEX 15)
    ERROR=.TRUE.
    RETURN
    END

```

C ***** SEEK *****

```

SUBROUTINE SEEK
IMPLICIT REAL*8 (A-H,O-Z)

```

CC PERFORMS A ROUGH MINIMIZATION BY MONTE CARLO SEARCH.

CC EACH TIME A NEW MINIMUM IS FOUND, THE SEARCH AREA IS SHIFTED

```

CC      TO BE CENTERED AT THE BEST VALUE.  RANDOM POINTS ARE CHOSEN
CC      UNIFORMLY OVER A HYPERCUBE DETERMINED BY CURRENT STEP SIZES.
CC
      REAL*4 RAN6
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /LIMITS/ ALIM(30) ,BLIM(30) ,LCODE(30) ,LCORSP(30) ,LIMSET
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON /DERIVA/ G(30) ,G2(30)
      DIMENSION AMID(15) , N(15)
      INTEGER CWORD,CWORD2,CWORD3
      WRITE (ISYSWR,1000)
      NUMBER = WORD7(1) + 0.5D0
      IF (NUMBER .LE. 0) NUMBER = 10 * NPAR
      IFLAG = 4
C—      INITIAL VALUES
      DO 100 J= 1, NU
      NI = LCORSP(J)
      IF (NI .LE. 0) GO TO 100
      N(NI) = J
      AMID(NI) = U(J)
100 CONTINUE
      NCYCL = 0
C      MONTE CARLO SEARCH OVER ENTIRE VARIABLE PARAMETER SPACE
      DO 650 INUM= 1, NUMBER
      DO 200 I2= 1, NPAR
      I3 = N(I2)
C      RANDOM POINTS IN UNIFORM DISTR.
180 XPLS=2.0D0*(DBLE(RAN6(0))-0.5D0)
      U(I3) = AMID(I2) + XPLS*WERR(I3)
      IF (LCODE(I3) .LE. 1) GO TO 200
      IF (U(I3) .GT. BLIM(I3) .OR. U(I3) .LT. ALIM(I3)) GO TO 180
200 CONTINUE
      CALL FCN(NPAR,G,F,U,IFLAG)
      NFCN = NFCN + 1
      IF (F .GE. AMIN) GO TO 650
      AMIN = F
      DO 500 I= 1, NPAR
      J = N(I)
500 AMID(I) = U(J)
      NCYCL = NCYCL + 1
      IF (ISW(5) .LT. 2) GO TO 650
      CALL EXTGIN(X)
      IF (ISW(5) .GE. 3 .OR. MOD(NCYCL,10) .EQ. 1) CALL MPRINT(0,AMIN)
650 CONTINUE
C      SEARCH FINISHED. SET U TO BEST VALUES
      DO 800 I= 1, NPAR
      NI = N(I)
800 U(NI) = AMID(I)
      CALL EXTGIN(X)
      WRITE (ISYSWR,1005)

```

```

      CALL MPRINT(1,AMIN)
      RETURN
1000 FORMAT ( 35H SEEK — MONTE CARLO MINIMUM SEARCH )
1005 FORMAT ( 28H0BEST VALUE FOUND IN SEEK IS )
      END
C *****SETALPH*****
      SUBROUTINE SETALPH (M)
      IMPLICIT REAL*8 (A-H,O-Z)
C
      INCLUDE 'BINFF.CCC'
      INCLUDE 'CDERIV.CCC'
C
      MALPH = MDAT(M,1)
      LOCALPH = INFF(10,MALPH) - 1
      NUMALPH = INFF(11,MALPH)
      H0 = DAT(5)
      RETURN
      END

C ***** SETCYB *****
      SUBROUTINE SETCYB(M,KA)
      IMPLICIT REAL*8 (A-H,O-Z)

C      SETS UP MILIST AND MATRIX FOR CYB INSTRUCTION

      INCLUDE 'CONTRL.CCC'
      INCLUDE 'BMI.CCC'

      DIMENSION RX(2,3),RY(2,3),RW(3),T(49)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      INCLUDE 'DIM.CCC'
      INTEGER do1STAR2/'** '/,do1BML/'BML '/,do1MMM/'MMM '/,
1      do1CYA/'CYA '/,do1CYC/'CYC '/,do1CELC/'CELC'/,
2      do1CFD/'CFD '/,do1DCFD/'DCFD'/,do1CELL/'CELL'/,do1ALTC/

      *'ALTC'/'
      KN = INFF(20,M)
      NT1 = MEND - 1
C      GET M OF INPUT ELEMENT AND ITS SYNCH OPNAME
      MPR = MDAT(M,1)
      NB = INFF(1,MPR)
      KNB = INFF(4,MPR)

      NGO=1
      GO TO 55
10  CONTINUE
      GO TO (11,15,20), ITP

```

```

C   IF IT IS A CELL TYPE, MATRIX IS IN STORAGE AND MI LIST IS AT LIN
11   IF (KA.GT.12) GO TO 65
      CALL DATA (MPR,3,1,12,MI)
      GO TO 30

C       MAKE MI LIST FROM BEAM LINE.
15   CONTINUE
      NDAT=INFF(17,MPR)
      CALL MIFILL(MPR,1,NDAT,KNB,MI)
      IF(KA.EQ.0) KA=KNB
      INFF(4,M)=KA
      IF (KA.GT.KNB) GO TO 65
      GO TO (21,30), IGO

C               FORM MATRIX FROM BEAM LINE
21   CALL MMM(NT1,KNB,MI)
      RETURN

20   CONTINUE
C   ** SYNCH INSTR. GET M, DETERMINE TYPE AND FORM MI LIST.
      MM = MDAT(MPR,1)
      NB = INFF(1,MM)
      NGO=2
      GO TO 55

25   CONTINUE
      IF (ITP.GT.2) GO TO 61
      GO TO (26,27), ITP

26   IF (KA.GT.12) GO TO 65
      CALL DATA (MM,3,1,12,MI)
      GO TO 30

27   CONTINUE
      KNB = INFF(4,MM)
      IF (KA.GT.KNB) GO TO 65
      CALL MLIST(MM,1,KNB,MI)

30   CONTINUE
C   IF KN IS 5, IT IS AN R7 MATRIX.
      IF (KN.EQ.5) GO TO 50

C   TRANSFER MATRIX FROM INDEX MPR TO WORKING STORAGE WITH INDEX MEND-1.
      CALL RXY(MPR,RX,RY,RW)
      CALL STXY (NT1,RX,RY,RW)
      RETURN

C   7 X 7 MATRIX
50   NT1 = M7END - 1
      CALL RTRV7(MPR,T,RW)
      CALL STOR7 (NT1,T,RW)
      RETURN

C*****

C       ANALYZE TYPE OF ELEMENT TO BE USED
55   IF ((NB.EQ.doICELL).OR.(NB.EQ.doIALTC)) GO TO 51
      IF ((NB.EQ.doICFD ).OR.(NB.EQ.doIDCFD)) GO TO 51
      IF (NB.NE.doICELC) GO TO 52

51   ITP = 1

```

```

      GO TO (10,25), NGO
52  IGO = 2
      IF ((NB.EQ.doICYA ).OR.(NB.EQ.doICYC )) GO TO 53
      IF (NB.EQ.doIMMM ) GO TO 53
      IF (NB.NE.doIBML ) GO TO 54
      IGO = 1
53  ITP = 2
      GO TO (10,25), NGO
54  IF (NB.NE.doISTAR2) GO TO 60
      ITP = 3
      GO TO (10,25), NGO

```

C*****

C*****

```

C          ERROR EXITS
60  WRITE (3,2) NB
2   FORMAT (5X,5H*** , 'CYB CANNOT USE A ',A5,' SYNCH INSTRUCTION. ')
      GO TO 70
61  WRITE (3,3)
3   FORMAT (5X,5H*** , ' CANNOT GET ELEMENTS FOR CYB.  STAR INSTR.
1   REFERS TO A BML OR  ANOTHER STAR INSTR. ')
      GO TO 70
65  WRITE (3,1) KA,NB,KNB
1   FORMAT (5X,5H*** , 'ERROR',1H*, 'CYB WANTS ',I5,' ELEMENTS FOR
1CYCLING.',A5,' HAS ONLY ',I5)
70  ERROR = .TRUE.
      RETURN
      END

```

C ***** SETFIT *****

```

      SUBROUTINE SETFIT
      IMPLICIT REAL*8 (A-H,O-Z)
C   SETS UP FIT INPUT DATA FOR FIT ROUTINE.
C   INPUT —  NAME  FITX   KA  KB  SUB  C  QF  QD  I1  I2  A1  A2
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'GRR.CCC'
      INCLUDE 'INSTR.CCC'
      INCLUDE 'STORE.CCC'

      DIMENSION MV(2),IV(2),Q(2)
      LOGICAL FSW
      INTEGER BDAT
      INTEGER doIFITB/4HFITB/,doIFITR/4HFITR/,doIFITQ/4HFITQ/,doIFITV/
*4HFITV/
      QSW = .FALSE.
      BSW = .FALSE.
      VSW = .FALSE.
      RSW = .FALSE.
      FSW = .FALSE.

```

```

C          TURN ON APPROPRIATE SWITCH
      IF (OPNAME.EQ. doIFITB)  BSW = .TRUE.
      IF (OPNAME.EQ. doIFITR)  RSW = .TRUE.
      IF (OPNAME.EQ. doIFITV)  VSW = .TRUE.
      IF (OPNAME.NE. doIFITQ) GO TO 100
      QSW = .TRUE.

```

```

C      IF KA NOT 0, TURN ON FSW TO TRIGGER CALL TO FITE (FOR FITQ ONLY)
      IF (KA.NE.0) FSW = .TRUE.
100    IF (MODE.EQ.3) GO TO 116

      IQ = 4
      IF (RSW) IQ =6
105    CALL RESRVA(M,0,0,IQ,2,2,1,2,3,0)
      CALL LOAD8(M,4,2,2,3,2,1,0)
      IF (.NOT.RSW) GO TO 110

C      IN FITR CASE, KA AND KB EACH CONTAIN 2 ELEMENT NUMBERS, WHICH ARE
C      STORED IN BCD INPUT TO BE DECODED LATER INTO INTEGERS.
      DECODE (8,1,ICARD(3)) K1,K2
1      FORMAT(A3,1X,A3,1X)
      CALL REPBCD(M,5,K1)
      CALL REPBCD(M, 6,K2)

110    IF (MODE.EQ.2) RETURN
C      XSW SUPPRESSES EXECUTION
115    IF (XSW) RETURN

116    CONTINUE
      MV(1)=BDAT(M,3)
      MV(2)=BDAT(M,4)
      IV(1)=IDAT(M,1)
      IV(2)=IDAT(M,2)
      CALL DATA(M,1,1,2,Q)
      MSR = MDAT(M,1)
      MHZ = MDAT(M,2)
      K1 = KA
      K2 = KB
      IF (.NOT.RSW) GO TO 117
      K1 = BDAT(M,5)
      K2 = BDAT(M,6 )
117    IF (FSW) GO TO 120

C      CALL FITT EXCEPT ON OPTION IN CASE OF FITQ.
      CALL FITT(MSR,MHZ,MV ,IV ,Q,K1,K2)
      RETURN
120    CALL FITE(MSR,MHZ,MV ,IV ,Q,K1,K2)
      RETURN
      END

C ***** SETMAT *****
      SUBROUTINE SETMAT(MT)
      IMPLICIT REAL*8 (A-H,O-Z)
C      ROUTINE SETS UP AND STORES UNIT AND TRANSPOSE MATRICES
      DIMENSION T(7,7),RW(3)
      ASSIGN 6 TO NEXT
      DO 3 I=1,3
3      RW(I)=0.
4      DO 5 J=1,7
      DO 5 I=1,7
5      T(I,J)=0.
      GO TO NEXT, (6,10)

```

C COMPUTE UNIT MATRIX NAMED (1)

```

6 DO 7 I=1,7
7 T(I,I)=1.
  RW(3) = T(5,6)
  CALL STOR7(MT,T,RW)
  MT=MT-1
  ASSIGN 10 TO NEXT
  GO TO 4

```

C COMPUTE TRANSPOSE NAMED (S)

```

10 DO 12 I=1,5,2
  T(I,I+1)=-1.
12 T(I+1,I)=1.
  T(7,7)=1.
  RW(3)=T(5,6)
  CALL STOR7(MT,T,RW)
  RETURN
END

```

C***** SETOPS *****

```

SUBROUTINE SETOPS
IMPLICIT REAL*8 (A-H,O-Z)

```

- * SETS UP LIST OF SYNCH INSTRUCTIONS.
- * OPNUM ROUTINE WILL USE INDEX TO SET OP NUMBERS

```

COMMON/OPLIST/LIST(200)
DATA IBLNK,IEQ/' ','=' '/'

```

```

DO 1 I=1,200
LIST(I) = IBLNK
1 CONTINUE

```

```

LIST(1) = 4HSTOP
LIST(2) = 4HFIN
LIST(3) = 4HRUN
LIST(4) = 4HSUB
LIST(5) = 4HEND
LIST(6) = 4HCALL
LIST(7) = 4HINCR
LIST(8) = 4HREPL
LIST(9) = 4HMESH
LIST(10) = 4HVPAR
LIST(11) = IEQ
LIST(12) = 4HCRD
LIST(13) = 4HBML
LIST(14) = 4HDRF
LIST(15) = 4HMAG
LIST(16) = 4HMAGV
LIST(17) = 4HEQU
LIST(18) = 4HINV
LIST(19) = 4HSHF
LIST(20) = 4HREF
LIST(21) = 4H**
LIST(22) = 4HMMM
LIST(23) = 4HCYA

```


LIST(24) = 4HCYB
LIST(25) = 4HCYC
LIST(26) = 4HCELL
LIST(27) = 4HALTC
LIST(28) = 4HTAB
LIST (29) = 4HPTAB
LIST(30) = 4HFITQ
LIST(31) = 4HFITB
LIST(32) = 4HFITR
LIST(33) = 4HFITV
LIST(34) = 4HPAGE
LIST(35) = 4HREM
LIST(36) = 4HWMMA
LIST(37) = 4HSIZE
LIST(38) = 4HWBE
LIST(39) = 4HFIT
LIST(40) = 4HCOPY
LIST(41) = 4HNCPY
LIST(42) = 4HBEST
LIST(43) = 3HBEP
LIST(44)=4HBETA
LIST(45)=4HKICK
LIST(46) = 4HRAND
LIST(47) = 4HSHF7
LIST(48) = 3HSOL
LIST(49) = 3HVAR
LIST(50) = 4HPRNT

LIST(51) = 4HSUM
LIST(52) = 4HACT
LIST(53) = 4HDELE
LIST(54) = 4HSCOP
LIST(55) = 4HTEST
LIST(56) = 4HWFL
LIST(57) = 4HMAGS
LIST(58) = 4HCELC
LIST(59) = 4HCFD
LIST(60) = 4HDCFD
LIST(61) = 4HSTR2
LIST(62) = 4HSTR4
LIST(63) = 4HSTRP
LIST(64) = 4HSTRN

LIST(65) = 4HFXPT
LIST(66) = 4HTRK
LIST(67) = 3HPRD
LIST(68) = 4HSXTP
LIST(69) = 3HMAP
LIST(70) = 4HBMIS
LIST(71) = 4HEMIS
LIST(72) = 4HROTZ
LIST(73) = 4HROT
LIST(74) = 4HINV2
LIST(75) = 4HELQ
LIST(78) = 4HMOVE

```
LIST(79) = 4HMOD
LIST(80) = 4HCON
LIST(81) = IBLNK
LIST(82) = 4HNPOL
LIST(83) = 4HORBC

LIST(100) = 4HPBML
LIST(101) = 4HMAT
LIST(102) = 4HVEC
LIST(103) = 4HPVEC
LIST(104) = 4HMXV
LIST(105) = 4HCPLX
LIST(106) = 4HEQIL
LIST(107) = 4HEVEC
LIST(108) = 4HPRTV
LIST(109) = 4HPRV7
LIST(110) = 4HCVEC
LIST(111) = 4HLIST
LIST(112) = 4HSIN
LIST(113) = 4HCOS
LIST(114) = 4HEXP
LIST(115) = 4HALOG
LIST(116) = 4HSQ
LIST(117) = 4HSQRT
LIST(118) = 4HTAN
LIST(119) = 4HASIN
LIST(120) = 4HACOS
LIST(121) = 4HATAN
LIST(122) = 4HABS
LIST(123) = 4HCALC
LIST(124) = 4HFBLE
LIST(125) = 4HFBLI
LIST(126) = 4HBMLE
LIST(127) = 4HBMLI
LIST(128) = 4HMAT7

LIST(151) = 4HBEAM
LIST(152) = 4HDKE
LIST(153) = 4HEDRF
LIST(155) = 4HCYEM
LIST(156) = 4HBVAL
LIST(157) = 4HCYAE
LIST(158) = 3HIBS
LIST(159) = 4HUPIO
LIST(160) = 4HPARA

C

LIST(161) = 4HTRKB
LIST(162) = 4HSOLV
LIST(163) = 4HSMIN
LIST(164) = 4HTRKE
LIST(165) = 4HTRKM
LIST(166) = 4HIBET
RETURN
END
```

```

C ***** SHF *****
      SUBROUTINE SHF(M,PAR)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BINFF.CCC'
      REAL*4 RAN6
      DIMENSION RX(2,3),RY(2,3),PAR(4),RW(3)
      DATA RW/3*0.0/
C  SETS UP A SHIFT MATRIX
      RX(1,1)=1.
      RX(1,2)=0.
      RX(1,3)=PAR(1)
      RX(2,1)=0.
      RX(2,2)=1.
      RX(2,3)=PAR(2)
      RY(1,1)=1.
      RY(1,2)=0.
      RY(1,3)=PAR(3)
      RY(2,1)=0.
      RY(2,2)=1.
      RY(2,3)=PAR(4)
      KB=INFF(5,M)
      IF(KB.LE.0) GO TO 2
C  IF(KB.GE.1) ERRORS ARE RANDOM WITH GIVEN RANGE
      DO1 K=1,2
      RX(K,3)=RX(K,3)*(DBLE(RAN6(0))-.5)
      1  RY(K,3)=RY(K,3)*(DBLE(RAN6(0))-.5)
      2  CONTINUE
      CALL STXY(M,RX,RY,RW)
      RETURN
      END
C ***** SHF7 *****
      SUBROUTINE SHF7(M,PAR)
      IMPLICIT REAL*8(A-H,O-Z)
C .....
C  SQ      SHF7          R17    R27    R37    R47    R57    R67
C .....
C  DEFINES A 7X7 SHIFT MATRIX, CONSISTING OF THE UNIT MATRIX WITH
C  THE 7TH COLUMN REPLACED BY THE INPUTED SIX NUMBERS.

      INCLUDE 'DIM.CCC'
      DIMENSION T(7,7),RW(3),PAR(6)

      CALL RTRV7(MUNIT,T,RW)
      DO 10 I=1,6
      10  T(I,7) = PAR(I)
      CALL STOR7(M,T,RW)
      RETURN
      END

C ***** SHIFT *****
      SUBROUTINE SHIFT(A, N, B)
      INTEGER SHFTR,SHFTL
      INTEGER A, B
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU, IDEBUG

```

```

      IF (IABS(N) .GE. 32) GO TO 30
      IF (N) 10,20,20
C     LEFT SHIFT
10    N = -N
      B = SHFTL(A, N)
      RETURN
C     RIGHT SHIFT
20    B = SHFTR(A, N)
      RETURN
C     ILLEGAL SHIFT
30    B = 0.
c$$$$ write(idebug, 1000) N
1000  FORMAT(' SHIFT COUNT > 32'. I6)
      RETURN
      END
C ***** SIMPLX *****
      SUBROUTINE SIMPLX
      IMPLICIT REAL*8 (A-H,O-Z)
CC     PERFORMS A MINIMIZATION USING THE SIMPLEX METHOD OF NELDER
CC     AND MEAD (REF. — COMP. J. 7,308 (1965)).
CC
      COMMON /PARINT/ X(15) ,XT(15) ,DIRIN(15) ,MAXINT ,NPAR
      INCLUDE 'PAREXT.CCC'
      COMMON /VARIAN/ V(15,15)
      COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
      COMMON /TITLE / TITLE(13) ,DATE(2) ,ISW(7) ,NBLOCK
      COMMON /CONVER/ EPSI ,APSI ,VTEST ,NSTEPQ ,NFCN ,NFCNMX
      COMMON /CARD / IDUM,CWORD ,CWORD2 ,CWORD3 ,WORD7(7)
      COMMON /MINIMA/ AMIN ,UP ,NEWMIN ,ITAU ,SIGMA
      COMMON /DERIVA/ G(30) ,G2(30)
      COMMON/CASC/ JH, JL, Y(16)
      COMMON /SIMVEC/ P(15,16),PSTAR(15) ,PSTST(15) ,PBAR(15) ,PRHO(15)
      DIMENSION XBEST(15)
      INTEGER CWORD,CWORD2,CWORD3
      LOGICAL CONVER
      DATA ALPHA,BETA,GAMMA,RHOMIN,RHOMAX/1.D0, .5D0,2.D0,4.D0,8.D0/
      IF (NPAR .LE. 0) RETURN
      NPFN=NFCN
      NPARP1=NPAR+1
      RHO1 = 1.0D0 + ALPHA
      RHO2 = RHO1 + ALPHA*GAMMA
      WG = 1.0D0/DFLOAT(NPAR)
      IFLAG=4
      WRITE(ISYSWR,100) EPSI
      DO 2 I= 1, NPAR
      IF (ISW(2) .GE. 1) DIRIN(I) = DSQRT(V(I,I)*UP)
      IF (DABS(DIRIN(I)) .LT. 1.0D-10*DABS(X(I))) DIRIN(I)=1.0D-8*X(I)
      IF (ITAU.LT. 1) V(I,I) = DIRIN(I)**2/UP
2 CONTINUE
      IF (ITAU .LT. 1) ISW(2) = 1
C**     CHOOSE THE INITIAL SIMPLEX USING SINGLE-PARAMETER SEARCHES
1 CONTINUE
      CONVER=.TRUE.
      YNPP1 = AMIN
      JL = NPARP1

```

```

Y(NPARP1) = AMIN
ABSMIN = AMIN
DO 10 I= 1, NPAR
  AMING = AMIN
  PBAR(I) = X(I)
  BESTX = X(I)
  KG = 0
  NS = 0
  NF = 0
4 X(I) = BESTX + DIRIN(I)
  CALL INTOEX(X)
  CALL FCN(NPAR,G, F, U, 4)
  NFCN = NFCN + 1
  IF (F .LE. AMING) GO TO 6
C    FAILURE
  IF (KG .EQ. 1) GO TO 8
  KG = -1
  NF = NF + 1
  DIRIN(I) = DIRIN(I) * (-0.400)
  IF (NF .LT. 3) GO TO 4
  NS = 6
C    SUCCESS
6 BESTX = X(I)
  DIRIN(I) = DIRIN(I) * 3.000
  AMING = F
  KG = 1
  NS = NS + 1
  IF (NS .LT. 6) GO TO 4
C    LOCAL MINIMUM FOUND IN ITH DIRECTION
8 Y(I) = AMING
  IF (AMING .LT. ABSMIN) JL = I
  IF (AMING .LT. ABSMIN) ABSMIN = AMING
  X(I) = BESTX
  XBEST(I)=BESTX
  DO 9 K= 1, NPAR
9 P(K,I) = X(K)
10 CONTINUE
  JH = NPARP1
  AMIN=Y(JL)
  CALL RAZZIA(YNPP1,PBAR)
  DO 20 I= 1, NPAR
20 X(I) = P(I,JL)
  CALL INTOEX(X)
  IF (ISW(5) .GE. 1) CALL MPRINT(0,AMIN)
  SIGMA = SIGMA * 10.00
  SIG2 = SIGMA
  IGNAL = 0
  NCYCL=0
C    . . . . . START MAIN LOOP
50 CONTINUE
  IF (IGNAL .GE. 10) GO TO 1
  IF (SIG2 .LT. EPSI .AND. SIGMA.LT.EPSI) GO TO 76
  SIG2 = SIGMA
  IF ((NFCN-NPFN) .GT. NFCNMX) GO TO 78
C    CALCULATE NEW POINT * BY REFLECTION

```

```

DO 60 I= 1, NPAR
PB = 0.D0
DO 59 J= 1, NPARP1
59 PB = PB + WG * P(I,J)
PBAR(I) = PB - WG * P(I,JH)
60 PSTAR(I)=(1.D0+ALPHA)*PBAR(I)-ALPHA*P(I,JH)
CALL INTOEX(PSTAR)
CALL FCN(NPAR,G,YSTAR,U,4)
NFCN=NFCN+1
IF(YSTAR.GE.AMIN) GO TO 70
C      POINT * BETTER THAN JL, CALCULATE NEW POINT **
DO 61 I=1,NPAR
61 PSTST(I)=GAMMA*PSTAR(I)+(1.D0-GAMMA)*PBAR(I)
CALL INTOEX(PSTST)
CALL FCN(NPAR,G,YSTST,U,4)
NFCN=NFCN+1
C      TRY A PARABOLA THROUGH PH, PSTAR, PSTST. MIN = PRHO
Y1 = (YSTAR-Y(JH)) * RHO2
Y2 = (YSTST-Y(JH)) * RHO1
RHO = 0.5D0 * (RHO2*Y1 -RHO1*Y2) / (Y1 -Y2)
IF (RHO .LT. RHOMIN) GO TO 66
IF (RHO .GT. RHOMAX) RHO = RHOMAX
DO 64 I= 1, NPAR
64 PRHO(I) = RHO*PSTAR(I) + (1.D0 -RHO)*P(I,JH)
CALL INTOEX(PRHO)
CALL FCN(NPAR, G, YRHO, U, 4)
NFCN = NFCN + 1
IF (YRHO .LT. Y(JL) .AND. YRHO .LT. YSTST) GO TO 65
IF (YSTST .LT. Y(JL)) GO TO 67
IF (YRHO .GT. Y(JL)) GO TO 66
C      ACCEPT MINIMUM POINT OF PARABOLA, PRHO
65 CALL RAZZIA (YRHO,PRHO)
IGNAL = MAX0(IGNAL-2, 0)
GO TO 68
66 IF (YSTST .LT. Y(JL)) GO TO 67
IGNAL = MAX0(IGNAL-1, 0)
CALL RAZZIA(YSTAR,PSTAR)
GO TO 68
67 IGNAL = MAX0(IGNAL-2, 0)
675 CALL RAZZIA(YSTST,PSTST)
68 NCYCL=NCYCL+1
IF (ISW(5) .LT. 2) GO TO 50
IF (ISW(5) .GE. 3 .OR. MOD(NCYCL, 10) .EQ. 0) CALL MPRINT(0,AMIN)
GO TO 50
C      POINT * IS NOT AS GOOD AS JL
70 IF (YSTAR .GE. Y(JH)) GO TO 73
JHOLD = JH
CALL RAZZIA(YSTAR,PSTAR)
IF (JHOLD .NE. JH) GO TO 50
C      CALCULATE NEW POINT **
73 DO 74 I=1,NPAR
74 PSTST(I)=BETA*P(I,JH)+(1.D0-BETA)*PBAR(I)
CALL INTOEX (PSTST)
CALL FCN(NPAR,G,YSTST,U,4)
NFCN=NFCN+1

```

```

      IF(YSTST.GT.Y(JH)) GO TO 1
C     POINT ** IS BETTER THAN JH
      IF (YSTST .LT. AMIN) GO TO 675
      IGNAL = IGNAL + 1
      CALL RAZZIA(YSTST,PSTST)
      GO TO 50
C
      . . . . . END MAIN LOOP
76 WRITE(ISYSWR,120)
      GO TO 80
78 WRITE(ISYSWR,130)
      ISW(1) = 1
      CONVER=.FALSE.
80 DO 82 I=1,NPAR
      PB = 0.D0
      DO 81 J=1,NPARP1
81 PB = PB + WG * P(I,J)
82 PBAR(I) = PB - WG * P(I,JH)
      CALL INTOEX(PBAR)
      CALL FCN(NPAR,G,YPBAR,U,IFLAG)
      NFCN=NFCN+1
      IF (YPBAR .LT. AMIN) CALL RAZZIA(YPBAR,PBAR)
      CALL INTOEX(X)
      IF (NFCNMX+NPFN-NFCN .LT. 3*NPAR) GO TO 90
      IF (SIGMA .GT. 2.0D0*EPSI) GO TO 1
90 IF(CONVER) GO TO 902
      CALL INTOEX(XBEST)
902 CALL MPRINT(1-ITAU, AMIN)
      RETURN
100 FORMAT(      37H START SIMPLEX MINIMIZATION      8X   65HCON
11VERGENCE CRITERION — ESTIMATED DISTANCE TO MINIMUM (EDM) .LT.E10.
22 )
120 FORMAT(1H ,  34HSIMPLEX MINIMIZATION HAS CONVERGED)
130 FORMAT(1H ,  38HSIMPLEX TERMINATES WITHOUT CONVERGENCE)
      END
C ***** SKIP1 *****
      SUBROUTINE SKIP1
C     SKIPS ONE PRINT LINE
      WRITE(3,1)
1   FORMAT(1H )
      RETURN
      END
C ***** SMIN *****
      SUBROUTINE SMIN(M)
      IMPLICIT REAL*8 (A-H,O-Z)
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (ISTORE(1),STORE(1))
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'MATCH2.CCC'

      INTEGER BDAT
C INSTRUCTION SMIN — SET COMMANDS FOR MINUIT

```

C INPUT HAS SAME STRUCTURE AS MINUIT COMMANDS, BUT SHIFTED TO START
 C IN COLUMN 21, WITH FIVE WORDS OF FLOATING POINT DATA PER COMMAND.
 C EXAMPLE

```
C FIT  SMIN   4      PRINTOUT   2.
C                      SEEK      500.
C                      SIMPLEX   1500.    0.1
C                      END RETURN
```

SMINCL = .TRUE.

KA=INFF(4,M)

DO 1 I=1,KA

DWORD(I) = BDAT(M,1+3*(I-1))

DWORD2(I)= BDAT(M,2+3*(I-1))

DWORD3(I)= BDAT(M,3+3*(I-1))

LFPI = 5*(I-1) + 1

CALL DATA(M,1,LFPI,5,DWORD7(1,I))

1 CONTINUE

RETURN

END

C ***** SOL *****

SUBROUTINE SOL(M,PAR)

IMPLICIT REAL*8 (A-H,O-Z)

C

C FORMS A MATRIX REPRESENTING A SOLENOID.

C

C S SOL L (BLANK) BRHO B0

C

C

C L = LENGTH OF SOLENOID

C BRHO = B*RH0 (THE RIGIDITY OF THE PARTICLE)

C B0 = LONGITUDINAL FIELD IN SOLENOID

C

REAL K,L

DIMENSION PAR(1),T(7,7),RW(3)

C

DO 11 J=1,7

DO 10 I=1,7

10 T(I,J) = 0.

11 CONTINUE

DO 12 I=1,3

12 RW(I) = 0.

C

L = PAR(1)

BRHO = PAR(3)

B0 = PAR(4)

K = B0/(2.*BRHO)

C = COS(K*L)

S = SIN(K*L)

SC = S*C

SSQ = S*S

CSQ = C*C

T(1,1) = CSQ


```

T(2,1) = -K*SC
T(3,1) = -SC
T(4,1) = K*SSQ
T(1,2) = SC/K
T(2,2) = CSQ
T(3,2) = -SSQ/K
T(4,2) = -SC
T(1,3) = SC
T(2,3) = -K*SSQ
T(3,3) = CSQ
T(4,3) = -K*SC
T(1,4) = SSQ/K
T(2,4) = SC
T(3,4) = SC/K
T(4,4) = CSQ
T(5,5) = 1.D0
T(6,6) = 1.D0
T(7,7) = 1.D0

```

C

```

RW(1) = L
CALL STOR7(M,T,RW)
RETURN
END

```

C***** SOLVE *****

```

SUBROUTINE SOLV(M)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

INCLUDE 'BINFF.CCC'
INCLUDE 'BMI1.CCC'
INCLUDE 'CONTRL.CCC'

```

```

INCLUDE 'MINU.CCC'
INCLUDE 'MATCH.CCC'
INCLUDE 'MATCH2.CCC'
INCLUDE 'MATCH3.CCC'

```

LOGICAL JWT

```

C DATA(NAMBFC(I),I=1,42)/4HNUX ,4HBX ,4HAX ,4HGX ,4HX ,
C 1 4HDX ,4HNUY ,4HBY ,4HAY ,4HGY ,4HY ,4HDY ,
C 2 4HS ,4HTHET ,
C 3 4HBXMX ,4HBYMX ,4HXXMAX ,4HQX ,4HQY ,4H ,4HGAMT ,
C 4 4HCHRX ,4HCHRY ,4HALPH ,4HYMAX ,4HBXMN ,4HBYMN ,4HXMIN ,
C 5 4HYMIN ,4HCIRC ,4HTHTX ,4HTHTY ,4H ,4HMTRX ,4HRES ,
C 6 4HRESN ,4HBXBY ,4HRND2 ,4HRND ,4HWST ,4HDXD ,4HAXAY /

```

C** DATA STATEMENT IS IN BLOCK DATA IN SYNCHINC(MATCHD) **

```

DIMENSION U0(30), NAIN(5),IND(5),NAI(5),NAMPAR(6),
1 BXI(9),BYI(9)
DIMENSION NAIPOS(6),NIP(2),IP(2),LIM1(2),LIM2(2)
INTEGER BDAT,ELNUM,OPTR
DATA M0/0/
DATA LIM1,LIM2/4HINIT,3HIAL,4H FI,3HNAL/
DATA MINZER/-32767/
DATA NPR/6/
DATA IBLNK/1H /

```

C

C INSTRUCTION SOLV — BETATRON FUNCTION FITTING USING MINUIT

C.....

C XX	SOLV	KA	KB//SR	TR	IP1	IP2	ITS	TOL	PRN	SAV	PRI	LAMDA
C (CONSTR CD	1)	BNAM	TYPE	POS1	POS2	POS3	POS4	POS5	POS6	BFIT		SIG
C (CONSTR CD	2)	BNAM	TYPE	POS1	POS2	POS3	POS4	POS5	POS6	BFIT		SIG

C (CONSTR CD	KA)	BNAM	TYPE	POS1	POS2	POS3	POS4	POS5	POS6	BFIT		SIG
C (VARIABLE CD	1)	PAR		I1	I2	I3	I4	I5	LOWER	UPPER	STP	
C (VARIABLE CD	2)	PAR		I1	I2	I3	I4	I5	LOWER	UPPER	STP	

C (VARIABLE CD	KB)	PAR		I1	I2	I3	I4	I5	LOWER	UPPER	STP	
----------------	-----	-----	--	----	----	----	----	----	-------	-------	-----	--

C.....

C 2ND FORMAT FOR VARIABLE CARDS —

C.....

C		PAR1	PAR1	PAR3	PAR4	PAR5		I	LOWER	UPPER	STOP
---	--	------	------	------	------	------	--	---	-------	-------	------

C.....

C PARK IS NAME OF KTH ELEMENT WHOSE ITH INPUT NUMBER IS TO BE VARIED..

C_____

C FIRST CARD —

C KA = NUMBER OF CONSTRAINT CARDS

C KB = NUMBER OF VARIABLE CARDS

C SR = NAME OF SYNCH SUBROUTINE WITH VARIABLES AND A TRKB INSTRUCTION

C TR = NAME OF TRKB INSTRUCTION IN SR.

C TR MAY ALSO BE A CYC INSTRUCTION, WITH ITS KA SET TO 21,

C OR AN ~~MM~~ OR EQUIVALENT INSTRUCTION THAT DEFINES A SINGLE

C TRANSFER MATRIX FOR THE LATTICE PERIOD.

C IP1 = START POSITION OF TRKB RUN

C IP2 = END POSITION OF TRKB RUN. IF ZERO, USE VALUES ON TRKB CARD.

C ITS = MAX NO. OF CALLS TO FCN BY MINUIT

C TOL = DESIRED TOLERANCE ON FCN (FCN .LE. 10**TOL)—AN INTEGER

C PRN = PRINTOUT LEVEL FROM MINUIT (0 GIVES LEAST OUTPUT)

C SAV——INITIALIZED TO 0 BY SRUN. IF SAV DIFFERENT FROM PREVIOUS

C VALUE, CURRENT VARIABLE VALUES ARE SAVED IN ARRAY V.

C SOLV ALWAYS TAKES INITIAL VARIABLE VALUES FROM V.

C NORMALLY SET SAV=0, THEN VALUES FROM PREVIOUS FIT ARE USED

C FOR STARTING VALUES. SET SAV=1 TO ALWAYS START FROM

C SAME INITIAL VALUES.

C PRI = 0 (OR BLANK) — PRINT BETAS BEFORE AND AFTER FITTING

C 1 — PRINT BETAS AFTER FITTING ONLY

C 2 — DO NOT PRINT BETAS

C 10,11,12 — SAME AS 0,1,2 AND FORCE DOIT TO ALWAYS CALL MIFILL

C LAMDA — SEE BELOW UNDER VARIABLES

C_____

C CONSTRAINT CARDS —

C		BNAM	TYPE	IP1	IP2	IP3	IP4	IP5	IP6	BFIT	SIGMA
---	--	------	------	-----	-----	-----	-----	-----	-----	------	-------

C EACH CONSTRAINT CARD MAY INVOLVE UP TO 6 SEPARATE POSITIONS

C BNAM = NAMES OF BETA FUNCTION CONSTRAINTS——

C		NUX	BX	AX	GX	X	Dx	NUY	BY	AY	GY	Y	DY
---	--	-----	----	----	----	---	----	-----	----	----	----	---	----

C		S	THET	——	WST	MATRIX
---	--	---	------	----	-----	--------

C

```

C   S = PATH LENGTH, THET = HORIZONTAL BEND ANGLE,
C   WST COMBINES THREE CONSTRAINTS AX, AY, AND X
C   MTRX ——— CONSTRAIN MATRIX ELEMENT I1,I2 OF MATRIX
C               CORRESPONDING TO TR.
C   GAMT ——— TRANSITION GAMMA ( TR MUST BE A CYC  21 INSTR.)
C
C   TYPE = STD OR BLANK ———BETAS SHOULD FIT SPECIFIED VALUES
C   TYPE = DIF ——— DIFFERENCE OF BETAS BETWEEN SUCCESSIVE POSITIONS
C       SUM ——— SUM          (DITTO)
C               SHOULD FIT SPECIFIED VALUE BFIT.
C       WHEN BFIT = S, DIF AFFORDS A WAY TO OBTAIN GIVEN SEPARATIONS
C       BETWEEN SPECIFIED POSITIONS.
C   TYPE = BFIT ——— BNAM IS THE NAME OF A PREVIOUSLY DEFINED MATRIX;
C               THE BETA FUNCTIONS OF THAT MATRIX ARE CALCULATED;
C               THEIR VALUES ARE THOSE THAT SOLV SEEKS TO FIT.
C   TYPE = NO ——— PASS OVER THIS CONSTRAINT
C
C   I1,I2, ...I5 ARE POSITIONS AT WHICH THE CONSTRAINT APPLIES.
C   THE POSITIONS CAN BE ENTERED SYMBOLICALLY IN THE FORM AAANN,
C   WHERE AAA IS THE NAME OF A BEAMLINE ELEMENT, AND NN DENOTES
C   WHICH OCCURRENCE OF AAA IN THE BEAMLINE IS MEANT.
C   IF NN IS BLANK, THE FIRST OCCURENCE OF THE NAME IS UNDERSTOOD.
C
C   SIGMA = GOODNESS OF FIT DESIRED (INVERSE WEIGHT).
C
C _____
C   IF BNAM IS NOT A RECOGNIZED NAME, CONSTRAINT IGNORED, BUT POSITIONS
C   CAN BE USED TO MARK FIT POINTS FOR SUBSEQUENT CONSTRAINT CARDS.
C   IF A CONSTRAINT CARD HAS NO POSITIONS, THE POSITIONS ON THE NEAREST
C   PREVIOUS CARD WITH POSITIONS PUNCHED WILL APPLY.
C
C   FCN = FUNCTION TO BE MINIMIZED BY SUBROUTINE FCN IS
C   FCN = SUM[( (VALUE-FITVALUE)/TOLERANCE)**2]/(NUMBER OF CONSTRAINTS)
C _____
C   VARIABLE CARDS ———
C   KB = NUMBER OF VARIABLE CARDS, EACH MAY HAVE 1 PARAMETER NAME
C       AND UP TO 5 PARAMETER INDICES.
C   SECOND FORMAT ———UP TO 5 PARAMETER NAMES FOLLOWED BY 1 INDEX.
C   IF A VARIABLE CARD HAS NO POSITIONS, THE POSITIONS ON THE NEAREST
C   PREVIOUS CARD WITH POSITIONS PRESENT WILL APPLY.
C
C   IF STEPSIZ NOT ENTERED, DEFAULT IS (UPRBOUND-LOWBOUND)/2.
C   IF STEPSIZ IS NEGATIVE, VARIABLE ENTERS INTO FCN WITH
C   FIT VALUE = (UPPER+LOWER)/2,  TOLERANCE = LAMDA*(UPPER-LOWER)/2
C _____
C   PASS ITS, TOL, PRN TO MINUIT
C       IF (SMINCAL) GO TO 5
C   SAVE DEFAULT VALUES
C       ISV1 = DWORD7(1,1)
C       ISV2 = DWORD7(1,2)
C       ISV3 = DWORD7(2,2)
C       ITS = IDAT(M,1)
C       IF(ITS.EQ.0) GO TO 3
C       DWORD7(1,2)=ITS

```

```

3  ITOL = IDAT(M,2)
   DWORD7(2,2)=10.D0**ITOL
4  IPR = IDAT(M,3)
   IF(IPR.EQ.0) GO TO 5
   DWORD7(1,1)=IPR
5  CONTINUE
   IVSAV0=IVSAV
   IVSAV = IDAT(M,4)
   IPP = IDAT(M,5)
   MIFLG = .FALSE.
   MATFLG = .FALSE.
   GLOBAL = .FALSE.
   IF (IPP.LT.10) GO TO 52
   MIFLG = .TRUE.
   IPP = IPP - 10
52 KAT=INFF(4,M)
   KAR=KAT
   KBT=INFF(5,M)
   KBR=KBT
   KBTS=KBT
   MSR=MDAT(M,1)
   MTR=MDAT(M,2)
   KATR = 0
   OPTR = INFF(1,MTR)
   IF ((OPTR.NE.4HTRKB).AND.(OPTR.NE.3HCYC)) GO TO 55
   MBML = MDAT(MTR,1)
   NDAT = INFF(17,MBML)
   CALL MIFILL(MBML,1,NDAT,NELS1,MI1)
   NELS2 = NELS1
   KBTR = INFF(5,MTR)
   IF (OPTR.NE.4HTRKB) GO TO 53
   KATR = INFF(4,MTR)
   CALL DATA (M,2,3,2,NIP)

   CALL LOCS(NIP,2,NELS1,MI1,IP)
   IP1 = IP(1)
   IP2 = IP(2)
   IF (IP1.EQ.0.AND.IP2.EQ.0) GO TO 54
   CALL REPINT(MTR,1,IP1)
   CALL REPINT(MTR,2,IP2)
   GO TO 54
53 IF (KBTR.LT.0) NELS2 = NELS1 + 1
54 LQTR = INFF(24,MTR)
   LTRTAB = LQTR - 14*KATR
   GO TO 51
55 MATFLG = .TRUE.
C 51 NASR = INFF(2,MSR)
51 CONTINUE
   XLAMDA=FDAT(M,1)

```

C GET CONSTRAINT DATA AND STORE IN ARRAYS FOR FCN

DO 61 KK=1,30

```

61  IPOS(7, KK)=0

      NCONSTR = 0
      K=0

      DO 2 N=1, KAR
      K=K+1
      ICON(K)=0
      NTYPE = BDAT(M, 8*N-2)
      ITYPE(K)=0
      IF(NTYPE.EQ.4HNO ) ITYPE(K)=-1
      IF(NTYPE.EQ.4HSTD ) ITYPE(K)=0
      IF(NTYPE.EQ.4HDIF ) ITYPE(K)=1
      IF (NTYPE.EQ.4HSUM ) ITYPE(K)=2
      NCONK = BDAT(M, 8*N-3)
      IF (NTYPE.NE.4HBFIT ) GO TO 62
      ITYPE(K)=3
      GO TO 63

62  DO 6 I=1, 42
6   IF (NCONK.EQ.NAMBFC(I)) ICON(K)=I

      ICONK = ICON(K)
      IF (ICON(K).GE.34.AND.ICON(K).LE.37) MIFLG = .TRUE.
      IF(ICON(K).EQ.0) ITYPE(K)=-1
63  LPOSK = 7 + 8*(N-1)
      IF (ICON(K).LE.14.OR.ICON(K).GE.34) GO TO 65
      IPOS(1, K) = NELS2
      NPOS(K) = 1
      GLOBAL = .TRUE.
      GO TO 8

65  CONTINUE
      CALL DATA (M, 2, LPOSK, 6, NAIPOS)
      CALL LOCS(NAIPOS, 6, NELS1, MI1, IPOS(1, K))
      NPOS(K)=0

      I=0
7   I=I+1
      IF (NAIPOS(I).NE.IBLNK) GO TO 72
      NP1=0
      I1=I

      DO 71 I1=I1, 6
      IPOS(I1, K) = IPOS(I1+1, K)
71  IF (NAIPOS(I).NE.IBLNK) NP1 = NP1 + 1

      I=I-1
      IF(NP1.EQ.0) GO TO 73
      GO TO 7

72  NPOS(K)=NPOS(K)+1
      IF(I.LT.6) GO TO 7

73  CONTINUE
      IF(NPOS(K).NE.0) GO TO 8

```

```

      IF (ICON(K).NE.0) GO TO 74
      K=K-1
      GO TO 2
74    CONTINUE
      IF (K.GT.1) GO TO 10
      NUMCON=1
      GO TO 32
10    NPOS(K)=NPOS(K-1)
      DO 9 I=1,6
9     IPOS(I,K)=IPOS(I,K-1)

8     CONTINUE
      NUMCON=NPOS(K)
32    CONTINUE
      IF (ITYPE(K).EQ.-1) NUMCON=0
      IF ( (ITYPE(K).EQ.1).OR.(ITYPE(K).EQ.2) ) NUMCON=NUMCON-1

      LFPK=2*(N-1)+1
      BET0(K) = FDAT(M,LFPK+1)
      SIG(K) = FDAT(M,LFPK+2)
      IF(SIG(K).EQ.0.) SIG(K)=1.D20
      WT(K)=1.D0/SIG(K)

      IF (ICON(K).LT.38) GO TO 13
      LL = 1
      IF (ICONK.NE.40.AND.ICONK.NE.39) GO TO 28
      LL = 2
      ICON(K) = 3
      ICON(K+1)=6
      ICON(K+2)=9

      IF (ICONK.EQ.40) GO TO 27
      LL = 3
      ICON(K+3) = 37
      BET0F = BET0(K)
      BET0(K) = 0.
      GO TO 27

28    IF (ICONK.NE.41) GO TO 29
      ICON(K) = 5
      ICON(K+1) = 6
      GO TO 27

29    IF (ICONK.NE.42.AND.ICONK.NE.38) GO TO 27
      ICON(K) = 3
      ICON(K+1) = 9

      IF (ICONK.EQ.42) GO TO 27
      LL = 2
      ICON(K+2) = 37
      BET0F = BET0(K)
      BET0(K) = 0.

27    CONTINUE

```

```

      KL=K
      DO 11 L=1,LL
      KL=KL+1
      ITYPE(KL)=ITYPE(K)
      NPOS(KL)=NPOS(K)
      BET0(KL)=BET0(K)
      SIG(KL)=SIG(K)
      WT(KL)=WT(K)
      DO 12 I=1,6
12    IPOS(I,KL)=IPOS(I,K)

11    CONTINUE

      IF (ICONK.EQ.38.OR.ICONK.EQ.39) BET0(KL)=BET0F
      NUMCON = (LL+1)*NUMCON
      K = KL
      KAT = KAT + LL
      GO TO 131

13    CONTINUE

      IF ((ICON(K).NE.35).AND.(ICON(K).NE.36)) GO TO 14
      NPOS(K) = 4
      NUMCON = 1
      IF (IPOS(3,K).EQ.0) IPOS(3,K)=1
      IF (IPOS(4,K).EQ.0) IPOS(4,K)=1
14    CONTINUE
      IF (ICON(K).NE.36) GO TO 24
      NN = IPOS(1,K)/2
      ISGN = 1
      IF (IPOS(2,K).LT.0) ISGN = -1
      IPOS(2,K) = 0
      ICON(K) = 35
      KK = K
      DO 25 II=1,NN
      KK = KK + 1
      DO 26 JJ=3,4
26    IPOS(JJ,KK) = IPOS(JJ,K)
      ICON(KK) = 35
      NPOS(KK) = 4
      SIG(KK) = SIG(K)
      BET0(KK) = BET0(K)
      WT(KK) = WT(K)
      IPOS(1,KK) = IPOS(1,KK-1) -2
      IPOS(2,KK) = IPOS(2,KK-1) + 2*ISGN
25    CONTINUE
      K = K + NN
      KAT = KAT + NN
      NUMCON = NN + 1
24    CONTINUE

      IF (ITYPE(K).NE.3) GO TO 131

```

```

MBET=ELNUM(NCONK)
CALL BET(MBET,BXI,BYI,0)
ICON(K )=2
ICON(K+1)=3
ICON(K+2)=5
ICON(K+3)=6
ICON(K+4)=8
ICON(K+5)=9
BET0(K )=BXI(2)
BET0(K+1)=BXI(3)
BET0(K+2)=BXI(5)
BET0(K+3)=BXI(6)
BET0(K+4)=BYI(2)
BET0(K+5)=BYI(3)

DO 132 L=1,5
  ITYPE(K+L)=ITYPE(K)
  NPOS(K+L)=NPOS(K)
  SIG(K+L)=SIG(K)
  WT(K+L)=WT(K)

DO 133 I=1,6
133  IPOS(I,K+L)=IPOS(I,K)

132  CONTINUE

      NUMCON=6*NUMCON
      K=K+5
      KAT=KAT+5
131  CONTINUE

      NCONSTR = NCONSTR + NUMCON
2    CONTINUE
      DO 82 K=1,30
      DO 82 I=1,6
82   IPRPOS(I,K) = IBLNK
      DO 90 K=1,NCONSTR
      NP=NPOS(K)
      IF (ICON(K).GT.14) GO TO 84
      IF ((OPTR.NE.4HTRKB).AND.(OPTR.NE.3HCYC)) GO TO 90
      GO TO 86
84   IF (ICON(K).LE.32) GO TO 90
      IF (ICON(K).LT.38) GO TO 86
      IF ((OPTR.NE.4HTRKB).AND.(OPTR.NE.3HCYC)) GO TO 90
86   DO 88 I=1,NP
88   ENCODE(4,106,IPRPOS(I,K)) IPOS(I,K)
106  FORMAT(I4)
90   CONTINUE

```

```

C  GET DATA ON VARIABLES AND LOAD COMMON BLOCKS FOR FCN AND MIDATA
C  MPAR(J) = M-NO OF JTH VARIABLE

```

```

VARWTS=.FALSE.
NVARWT=0

```



```

IF(XLAMDA.EQ.0.) XLAMDA=1.D0
L=0

DO 1 J=1,KBR
  L2=1
  LOCJ=2*KAR+3*J-2
  AAJ=FDAT(M,LOCJ+1)
  BBJ=FDAT(M,LOCJ+2)
  WRJ=FDAT(M,LOCJ+3)
  VALJ=(BBJ+AAJ)/2.D0
  BBJJ=BBJ-AAJ
  DEL=DABS(BBJJ)/2.D0
  IF (DEL.EQ.0.) DEL=1.D0
  SIGV=XLAMDA*DEL
  WTVARJ=1.D0/SIGV
  JWT=.FALSE.
  IF(WRJ.EQ.0.) WRJ=DEL
  IF(WRJ.GE.0.) GO TO 23
  VARWTS=.TRUE.
  JWT=.TRUE.
  WRJ=-WRJ
23  CONTINUE
  LVARJ = 8*KAR + 6*J -1
  MPARJ=MDAT(M,LVARJ)
  MPAR(L+1)=MPARJ
  NAMPAR(1)=BDAT(M,LVARJ)

C  FIND OUT FORMAT TYPE
  IFORM=1
  LI=L+1

  DO 17 I=1,5
    ERPR=.FALSE.
    MPARI=MDAT(M,LVARJ+I)
    IF (MPARI.EQ.MINZER) GO TO 16
    IFORM=2
    L2=L2+1
    LI=LI+1
    MPAR(LI)=MPARI
    NAMPAR(L2)=BDAT(M,LVARJ+I)
    GO TO 17
16  NAIND(I)=BDAT(M,LVARJ+I)
    DECODE(4,100,NAIND(I)) IND(I)
100  FORMAT(I4)
    IF(IND(I).EQ.0) GO TO 17
    NAIND2=NAIND(I)
    IND2=IND(I)
17  CONTINUE

  L0=L+1
  IF(IFORM.EQ.2) GO TO 18
C  FORMAT 1

```

```

DO 15 I=1,5
  IF(IND(I).EQ.0) GO TO 15
  NEWORD = NAIND(I)
  L=L+1
  NAM(L)=NAMPAR(1)
  NMIND(L) = NEWORD
  MPAR(L)=MPARJ
  IPAR(L)=IND(I)
15  CONTINUE
C  IF NO INDEX WAS PRESENT, USE PREVIOUSLY DEFINED ONE.
  IF (L.GE.L0) GO TO 20
  L = L + 1
  NEWORD = NAIND2
  NAM(L) = NAMPAR(1)
  NMIND(L) = NEWORD
  MPAR(L) = MPARJ
  IPAR(L) = IND2
  GO TO 20

C  FORMAT 2
18  LL2=L2

DO 19 L2=1,LL2
  L=L+1
  NEWORD = NAIND2
  NAM(L) = NAMPAR(L2)
  NMIND(L) = NEWORD
  IPAR(L)=IND2
19  CONTINUE

20  L1=L

DO 21 L=L0,L1
  U(L)=FDAT(MPAR(L),IPAR(L))
  IF ((IVSAV.NE.0).AND.(IVSAV.EQ.IVSAV0).AND.(M.EQ.M0)) GO TO 22
  U0(L)=U(L)
22  U(L)=U0(L)
  AA(L)=AAJ
  BB(L)=BBJ
  WERR(L)=WRJ
  LVAR(L)=.FALSE.
  WTVAR(L)=0.
  IF(.NOT.JWT) GO TO 21
  LVAR(L)=.TRUE.
  WTVAR(L)=WTVARJ
  VAL(L)=VALJ
  NVARWT=NVARWT+1
21  CONTINUE
C
  L = L1

  L = L1
1  CONTINUE

NCONTOT=NCONSTR+NVARWT

```

```

      FNORM=1.D0/DFLOAT(NCONTOT)
      M0=M
      NPAR=L
      KBT=L
      KBTS=KBT
      CALL DASH
      WRITE (3,102)
      WRITE (3,105) LIM1,(I,NAM(I),NMIND(I),U(I),I=1,NPAR)
      CALL FCN(NPAR,G,AMIN,U,1)
      CALL DASH

      CALL MINNEW

      WRITE (3,107) LIM2,(I,NAM(I),NMIND(I),U(I),I=1,NPAR)
      CALL FCN(NPAR,G,F,U,3)
      MINFLG=1
      CALL DASH
C  RESTORE DEFAULTS IF SAVED
      MIFLG = .FALSE.
      NOPR = .FALSE.
      IF (SMINCAL) RETURN
      DWORD7(1,1) = ISV1
      DWORD7(1,2) = ISV2
      DWORD7(2,2) = ISV3
      RETURN
102  FORMAT (10X,'—— SOLV —— BETA-FUNCTION FITTING ')
105  FORMAT (1H0,5X,A4,A4,' VALUES OF VARIABLES'/
1      (7X,I5,3X,2A5,5X,E15.8))
107  FORMAT (1X,130(1H-),/6X,A4,A4,' VALUES OF VARIABLES'/
1      (7X,I5,3X,2A5,5X,E15.8))

      END

C ***** SPLOAD *****
      SUBROUTINE SPLOAD
      IMPLICIT REAL*8 (A-H,O-Z)
C  SPECIAL LOAD ROUTINE
C  INPUT HAS P OR C OR . IN COL. 1, FOLLOWED BY HOLLERITH DATA
C  CONSTRUCTS A PAGE OR REM SYNCH INSTRUCTION
C  IF NOT IN MODE 2, EXECUTES PAGE AND/OR REMARK
C  WHEN IN MODE 3, IT IS HANDLED NORMALLY BY SWITCH

      DIMENSION ICOM(20)
      INCLUDE 'INSTR.CCC'
      INCLUDE 'STORE.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'COPY.C.C'

C  LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF
      INTEGER MASK1 /ZFFFFFF00/
      INTEGER MASK2 /Z000000AA/
      DATA IB/' '/

      IF (ISIGN.EQ.1HP) OPNAME=4HPAGE
      IF (ISIGN.EQ.1HC) OPNAME=3HREM

```

```

IF (ISIGN.EQ.1H.) OPNAME=3HREM
CALL STINFO(M,IB,OPNAME,0,0,0)
DECODE(80,2,ICARD)(ICOM(I),I=1,20)
INS = IB

```

```

C          RESERVE AND STORE
CALL RESRV6(M,0,0,20,2,0)
CALL STDAT(M,2,1,20,ICOM)
IF (MODE.EQ.2) GO TO 10
IF (.NOT.CPYSW) RETURN
IF (MODE.EQ.1) GO TO 10
IF (ISIGN.EQ.1HP) INS =ISIGN
GO TO 12

10  IF (ISIGN.EQ.1HP) WRITE (3,1)
12  WRITE (3,3) INS,(ICOM(I),I=1,20)
    RETURN

1   FORMAT (1H1)
2   FORMAT (1X,3A4,A3)
3   FORMAT (4X,A1,1X,20A4)

```

END

```

C ..... SRUN .....
SUBROUTINE SRUN(IENTRY)
IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'INSTR.CCC'
INCLUDE 'CONTRL.CCC'
INCLUDE 'DIM.CCC'
INCLUDE 'FLTN.CCC'
INCLUDE 'STORE.CCC'
INCLUDE 'GRR.CCC'
INCLUDE 'SWTCH.CCC'
INCLUDE 'SVNAM.CCC'
INCLUDE 'TRKINT.CCC'
INCLUDE 'PLTIN.CCC'
INCLUDE 'MATCH.CCC'
INCLUDE 'VERSION.CCC'
DATA IBLNK/1H /
INTEGER DIF
INTEGER MXPMY /1/.R7/5/
INTEGER NEGZER / -32767 /
C SYNCHINC(VERSION) IS COMMON/VERSION/CODE(2),ID,VER
C CODE IS MACHINE CODE ID—VAX OR CDC
C SET ID TO 1 OR 2
C VER IS VERSION NUMBER BY YEAR AND MONTH
    DIMENSION INAM(2)
    DATA INAM/3H(1),3H(S)/
    EQUIVALENCE (MIN,MS)

```

C

```

      IF (OP.LT.0) RETURN
C     GO TO (RUN, FIN)
      GO TO (4,5), IENTRY
4     CONTINUE
C     CALL DUPIN TO READ INPUT AND WRITE TO ANOTHER FILE
C     THEN FILE IS REPOSITIONED TO CONTINUE NORMAL EXECUTION.
      CALL DUPIN
      PLTIN = .FALSE.
      WRITE (3,100) CODE(ID),VER
      WRITE (3,1) NAME,(ICARD(J),J=6,20)
      DO 50 I=1,15
50    NRUNCOM(I) = ICARD(I+5)
      CALL DATE(BUF)
      CALL TIME(BUFF)
      WRITE (3,101) BUF,BUFF
C     PUT ICARD INTO ITITLE FOR PLOT ROUTINES
      DO 6 J=1,15
6     ITITLE(J) = ICARD(J+5)
C     ISAV AND ISAV7 ARE NUMBER OF INFOS RESERVED FOR WORKING STORAGE
1     FORMAT (8X,15H      SYNCH RUN ,A5,10X,15A4)
C     LEND = LMAX = DIMENSION OF STORE
C     MEND = MAX = DIMENSION OF M IN INFO
C     RESERVE WORKING SPACE FOR MATRICES AND LENGTH
C     ZERO STORAGE
      DO 25 I=1,LMAX
25    STORE(I)=0.
      DO 26 I=1,MAX
      DO 26 J=1,24
26    INFF(J,I)=0
      EMPTY=.FALSE.
      INDEF=.FALSE.
      XEQ=.TRUE.
      STOR=.TRUE.
      RSRV=.TRUE.
      ERROR=.FALSE.
      CYCSWT=.FALSE.
      IERR=.FALSE.
      PV = .FALSE.
      MSSW = .FALSE.
      VPR = .FALSE.
      MHS=.FALSE.
      MODS=.FALSE.
      ERPR=.TRUE.
      IVSAV=0
      MODE=1
      NF=0
      NB=0
      NI=0
      NAMRUN=NAME
      BEND=1.
      LFILE=1
      MS=MEND
      DO 3 I=1,ISAV
      CALL RESRV4(MS,MXPMY,NEGZER,0)
      INFF(1,MS) = IBLNK

```

```

      INFF(2,MS) = IBLNK
      MS=MS-1
3     CONTINUE
      MSIZE = 37
      I7 = ISAV7 - 2
      IF (ISAV7.EQ.0) GO TO 7
      DO 10 I=1,I7
      CALL RESRV4(MS,R7,NEGZER,0)
      INFF (1,MS) = IBLNK
      INFF(2,MS) = IBLNK
      MS = MS - 1
10    CONTINUE
C   SET UP INFO AND STORAGE FOR UNIT AND TRANSPOSE MATRICES
7     CONTINUE
      MT = MS
      MUNIT = MS
      DO 15 I=1,2
      INFF(1,MS) = IBLNK
      INFF(2,MS) = INAM(I)
      CALL RESRV4(MS,R7,NEGZER,0)
15    MS = MS - 1
      MSYMP1 = MS
C   DEFINE AND STORE MATRICES
      CALL SETMAT(MT)
C   RESTORE INFO FOR RUN
      CALL WRTINF(OPNAME,MS)
      GO TO 8
C
C   END OF RUN
5     CONTINUE
      EMPTY=.FALSE.
      FIN = .TRUE.
      NSTO = LMAX - LFILE
      NINFF = MAX - MIN
      WRITE (3,11) NAME,OPNAME,KA,KB,LMAX,LFILE,NSTO,MAX,NINFF,MIN
11    FORMAT (/6H *** ,1X,A5,2X,A5,1X,I3,1X,I3,1X,3H //,1X,
2     'CORE USE SUMMARY',27X,'MAXIMUM',17X,'USED',8X,'UNUSED'/
3     34X,'STORE (ELEMENT STORAGE)',14X,I10,1X,'(LMAX)',4X,I10,5X,I10/
4     34X,'INFF (ELEMENT DEFINITIONS)',10X,I10,1X,'(MAX)',5X,I10,5X,
5     I10)
      WRITE (3,2) NAMRUN
2     FORMAT(/1X,130(1H=)/18H END OF SYNCH RUN ,A5)
100   FORMAT (1H1,99X,'SYNCH VERSION 'A4,A4)
101   FORMAT (13X,A9,2X,A8//1X,130(1H=))
8     RETURN
      END

C ***** STAR *****
      SUBROUTINE STAR(M,KA)
      IMPLICIT REAL*8 (A-H,O-Z)
C   THE MATRIX M = MATRIX N RAISED TO POWER K (M=N**K)
      INCLUDE 'BMI.CCC'

      INTEGER BDAT
      N=MDAT(M,1)

```

```

DO 20 I=1,KA
MI( I )=N
20 CONTINUE
CALL MMM(M,KA,MI)
RETURN
END

C ***** STDAT *****
SUBROUTINE STDAT(M,I,J,K,A)
IMPLICIT REAL*8 (A-H,O-Z)
C REPLACES K DATA OF TYPE I FROM ARRAY A INTO INFF(J,M)
C IF I=1, FL. PT. IF I=2, BCD. IF I=3, INTEGER.
C IF I=4, SKIP. IF I=5, STORE AT LQ
C IF I=7, STORE AT LQ2. IF I=8, STORE AT LQ3
INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
EQUIVALENCE (A1,IA(1))
DIMENSION A(1),LOC(10),IA(2)
INCLUDE 'STORE.CCC'

IF (I.EQ.4) GO TO 20
LOC(1) = INFF(14,M)
LOC(2)=INFF(16,M)
LOC(3)=INFF(12,M)
LOC(5) = INFF(10,M)
LOC(7) = INFF(19,M)
LOC(8)=INFF(24,M)
IF (I .EQ. 2 .OR. I .EQ. 3) GO TO 15
LOC1 = LOC(1)+J-2
KMAX = LMAX + K
IF (LOC1 .GT. 0 .AND. LOC1 .LE. KMAX) GO TO 5
GO TO 20
5 CONTINUE
C FLOATING POINT
DO 10 L=1,K
LOC1 = LOC1+1
10 STORE(LOC1) = A(L)
GO TO 20
C BCD OR INTEGER
15 CONTINUE
LOC1 = LOC(1)*2 -1
LOC1 = LOC1 +J -2
K1 = (K+1) / 2
KK = 0
DO 16 L = 1,K1
LOC1 = LOC1 +1
A1 = A(L)
IIA = IA(1)
ISTORE(LOC1) = IIA
KK = KK + 1
IF (KK.GE.K) GO TO 16
LOC1 = LOC1 +1
IIA = IA(2)

```

```

        ISTORE(LOC1) = IIA
        KK = KK + 1
16      CONTINUE
        20 CONTINUE
        RETURN
        END

```

C ***** STINFO *****

```

        SUBROUTINE STINFO( M0,NAME0,OPNAM0,KA0,KB0,NAME1 )
        IMPLICIT REAL*8 (A-H,O-Z)
C      STORES NAME, OPNAME, OP, KA, KB  IN INFO(M0)
C      EXAMPLE----- CALL STINFO( MH,5HROVER,5HMM   .6,0,5HFIDO )
C      ----- NAME IS ROVER.
C      CALL STINFO(-MH,4HOX   ,5HMM   ,6,0,5HFIDO )
C      ----- NAME IS FOX.
C      IF M0 IS +, NAME IS  NAME0.  IF M0 IS -, NAME IS NAME0 PRECEDED BY
C      THE FIRST LETTER OF NAME1.
        INCLUDE 'BINFF.CCC'
        INTEGER OPNUM,OPNAM0
        DATA MASK1/ZFF000000/
        IF (M0) 2,1,1
1       INFF(2,M0) = NAME0
        GO TO 3
2       M0= -M0
        INFF(2,M0)=LAND(NAME1,MASK1)+SHFTR(NAME0,8)
3       INFF(1,M0) = OPNAM0
        INFF(3,M0) = OPNUM(OPNAM0)
        INFF(4,M0) = KA0
        INFF(5,M0) = KB0
        RETURN
        END

```

C *****STKLFT*****

```

        SUBROUTINE STKLFT(X,Y,Z,T)
        IMPLICIT REAL*8 (A-H,Z)

        T = Z
        Z = Y
        Y = X
        RETURN
        END

```

C *****STOREC*****

```

        SUBROUTINE STOREC(M,I,NREG,NSY)
        IMPLICIT REAL*8 (A-H,O-Z)

        INCLUDE 'BSTORE.CCC'
        INCLUDE 'BINFF.CCC'
        INCLUDE 'BWORK.CCC'
        INTEGER BDAT

        DECODE (4,50,IREG) LC
50      FORMAT (A1,3X)
        IREG = BDAT(M,I+1)
        IF (LC.EQ.1H ) GO TO 1
        NREG = -1
        MN = MDAT(M,I+1)
        NSY = INFF(10,MN)

```



```

      RETURN
1    DECODE(4,2,IREG) NR
2    FORMAT(I4)
      NREG = NR + 10
      RETURN
      END
C ***** STOR7 *****
      SUBROUTINE STOR7(M,T,RW)
      IMPLICIT REAL*8 (A-H,O-Z)
C    STORES 7X7 MATRICES
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'DIM.CCC'

      DIMENSION T(49),RW(3),TINV(49)
      DIMENSION RX(2,3), RY(2,3), RPQ(4)
      INTEGER doIMXY /'MXY '/
      IF (M) 4,10,10
4     M=-M
      CALL TINV7(TINV,T)
      RW(1)=-RW(1)
      RW(2)=-RW(2)
      RW(3)=TINV(40)
      DO 5 I=1,49
        T(I)=TINV(I)
5     CONTINUE
10    LQ=INFF(10,M)
      NQ=INFF(11,M)
      KIND = INFF(20,M)
      IF (KIND .EQ. doIMXY) GO TO 11
C    CHECK STORAGE
      NQQ=49+KADD
      IF (NQQ .GT. NQ) GO TO 45
      CALL STDAT (M,5,1,3,RW)
      NJ= LQ+KADD-1
      DO 20 I=1,49
        STORE(NJ+I)=T(I)
20    CONTINUE
      GO TO 50
45    MN= INFF(2,M)
      PRINT 1, MN, NQQ, NQ, M
      ERROR = .TRUE.
1    FORMAT (' *** ERROR *** STORAGE REQUEST EXCEEDS STORAGE RESERVE',
1 ' FOR ',A6, 3I6)
      GO TO 50
11    CONTINUE
      CALL CNVMAT(3, T, RX, RY, RW, RPQ)
      CALL STXY(M, RX, RY, RW)
50    CONTINUE
c$$$$ write(idebug, 1001)
1001 FORMAT(' STOR7 END')
      RETURN
      END
C ***** STR4 *****

```

```

SUBROUTINE STR4(M,NCELL,PAR)
  IMPLICIT REAL*8 (A-H,O-Z)
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))
  INCLUDE 'GRR.CCC'
  RETURN
  END

```

C ***** STXY *****

```

SUBROUTINE STXY(M,RX,RY,RW)
  IMPLICIT REAL*8 (A-H,O-Z)

```

C STORES RX AND RY MATRICES IN LQ(M). DIMENSIONS ARE (2,3) FOR EACH.

C IF ELEMENT NUMBER IS -, STORE INVERSE

```

  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  DIMENSION ISTORE(1)
  EQUIVALENCE (STORE(1),ISTORE(1))
  INCLUDE 'CONTRL.CCC'
  INCLUDE 'SWTCH.CCC'
  INCLUDE 'DIM.CCC'

```

```

  DIMENSION RX(2,3),RY(2,3),AX(2,3),AY(2,3),RW(3)
  DIMENSION RPQ(4),T(49)
  EQUIVALENCE (P1,RPQ(1)),(P2,RPQ(2)),(Q1,RPQ(3)),(Q2,RPQ(4))
  INTEGER doIMXY/'MXY' '/'

```

C

```

  IF (M.GE.0) GO TO 7

```

4

```

  M = - M
  AX(2,1) = -RX(2,1)
  AX(1,1) = RX(2,2)
  AX(2,2) = RX(1,1)
  AX(1,2) = -RX(1,2)
  AX(1,3) = RX(1,2) * RX(2,3) - RX(2,2)*RX(1,3)
  AX(2,3) = RX(2,1) *RX(1,3) - RX(1,1)*RX(2,3)
  AY(1,1) = RY(2,2)
  AY(2,2) = RY(1,1)
  AY(1,2) = - RY(1,2)
  AY(2,1) = - RY(2,1)
  AY(1,3) = RY(1,2)*RY(2,3) - RY(2,2)*RY(1,3)
  AY(2,3) = RY(2,1)*RY(1,3) - RY(1,1)*RY(2,3)
  DO 5 J=1,3
  DO 5 I=1,2
  RX(I,J) = AX(I,J)
  RY(I,J) = AY(I,J)

```

5

```

  CONTINUE
  RW(1) = -RW(1)
  RW(2) = -RW(2)
  IF (BEND.NE.0.) RW(3) = -RW(3)

```

7

```

  CONTINUE
  P1 = RX(1,3)*RX(2,1) - RX(2,3)*RX(1,1)
  P2 = RX(1,3)*RX(2,2) - RX(2,3)*RX(1,2)
  Q1 = RY(1,3)*RY(2,1) - RY(2,3)*RY(1,1)

```

```

Q2 = RY(1,3)*RY(2,2) - RY(2,3)*RY(1,2)
LQ = INFF(10,M)
NQ = INFF(11,M)
KIND = INFF(20,M)
IF (KIND.NE.doIMXY) GO TO 11

```

C CHECK STORAGE REQUIREMENT

```

      NNQ = 12 + KADDR
      IF (NNQ.GT.NQ) GO TO 20
      NJ = LQ + KADDR
      NI = NJ + 6
      DO 10 IJ=1,3
      DO 10 II=1,2
      STORE(NJ)=RX(II,IJ)
      STORE(NI)=RY(II,IJ)
      NJ=NJ+1
      NI=NI+1
10    CONTINUE
      CALL STDAT(M,5,1,3,RW)
      CALL STDAT(M,5,4,4,RPQ)
      GO TO 12

```

20 WRITE (3,1) M, NNQ, NQ

```

1  FORMAT (' *** ERROR *** INSUFFICIENT STORAGE RESERVED AT INDEX',
13I5)
      ERROR=.TRUE.
      GO TO 12

```

```

11  CALL CNVMAT(7,T,RX,RY,RW,RPQ)
      CALL STOR7(M,T,RW)

```

```

12  CONTINUE
      RETURN
      END

```

C ***** SUBR *****

```

      SUBROUTINE SUBR(IENTRY)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

C      SUBROUTINE LOGIC. INSTRUCTION /Q /SUB /... MEANS Q IS THE NAME
C      OF A S/R CONSISTING OF INSTRUCTIONS FOLLOWING UP TO AN END
C      INSTRUCTION. Q MAY LATER BE REQUESTED BY... / /CALL/K/ /Q .

```

```

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      DIMENSION ISTORE(1)
      EQUIVALENCE (STORE(1),ISTORE(1))
      INCLUDE 'INSTR.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'SWTCH.CCC'
      INCLUDE 'STORE.CCC'
      INCLUDE 'COPY.CCC'

```

C LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF

```

      LOGICAL MOSS
      INTEGER BDAT,ELNUM
      INTEGER doIIZ / 0 /
      DIMENSION INSUB(10),ITNO(10)

```

LOGICAL XSW

XSW = .FALSE.
 IF (OP.LT.0) XSW = .TRUE.
 MOSS = .FALSE.

C

EMPTY=.FALSE.
 IF(IENTRY.EQ.3) EMPTY=.TRUE.
 GO TO (60,70,80),IENTRY

60 CONTINUE

IF(.NOT.CPYSW) GO TO 50
 WRITE (3,51) NAME,OPNAME,KA,KB,(ICARD(IW),IW=6,20)
 CALL HED

50 IF(MODE.EQ.1) GO TO 1

ERROR=.TRUE.
 WRITE (3,52)
 GO TO 83

51 FORMAT (/6H *** ,1X,A5,2X,A5,1X,I3,1X,I3,5H // ,15A4)

52 FORMAT ('ERROR***SUB INSTRUCTIONS MUST NOT BE NESTED DURING',
 1'DEFINITION')

54 FORMAT (6H *** ,1X,A5,2X,A5,1X,I3,1X,I3,5H // ,15A4)

1 MODE =2
 XEQ=.FALSE.
 MSUBR=M
 CALL RESRV6(M,0,0,4,3,0)
 GO TO 83

C END ENTRY

70 CONTINUE
 GO TO (4,2,3),MODE

4 CONTINUE
 IF (MSSW) GO TO 25
 ERROR = .TRUE.
 WRITE (3,53)
 GO TO 83

53 FORMAT (70H ERROR****INSTRUCTION NAMED END MUST BE PRECEDED BY A S
 1UB INSTRUCTION.)

C SET TRA TO MSUBR

2 CALL INFW(MSUBR,9,M)
 IF(.NOT.CPYSW) GO TO 56
 WRITE (3,54) NAME,OPNAME,KA,KB,(ICARD(IW),IW=6,20)
 CALL HED
 CALL SKIP1
 56 MODE=1
 XEQ=.TRUE.
 GO TO 83

C MOSS CONTROLS MODE TO ALLOW PROPER FLOW OF PROGRAM WHEN MESH OR

C VPAR ARE OPERATIVE.

25 MOSS = .TRUE.
 3 MSUBR = INFF(9,M)
 KT = IDAT(MSUBR,3)
 ICOUNT = IDAT(MSUBR,4)
 ICOUNT = ICOUNT -1
 KT = KT + 1
 CALL REPINT(MSUBR,3,KT)
 CALL REPINT(MSUBR,4,ICOUNT)

```

CALL REPINT(MSUBR,1,0)
CALL REPINT(MSUBR,2,0)
NAMESR=MNAME(MSUBR)
IF(ICOUNT-1)5,6,7
5  MCALL = INFF(9,MSUBR)
   ISNB=ISNB-1
   M=MCALL
   IF (MOSS) GO TO 8
   IF(MODS) MODE=1
   IF(.NOT.MODS) MODE=INFF(19,MCALL)
   GO TO (8,9,10),MODE
8  XEQ=.TRUE.
   STOR=.TRUE.
   RSRV=.TRUE.
   GO TO 83
9  STOR=.TRUE.
   RSRV=.TRUE.
   XEQ=.FALSE.
   GO TO 83
10 STOR=.FALSE.
   RSRV=.FALSE.
   XEQ=.TRUE.
   GO TO 83
6  CALL REPINT(MSUBR,2,1)
7  TRA=MSUBR-1
   TRASW=.TRUE.
   DO 20 IS=1,10
     IST = IS
     IF (INSUB(IS).EQ.NAMESR) GO TO 21
20  CONTINUE
C  ERROR HERE AS NAME SHOULD MATCH.  PRINT MESSAGE AND CONTINUE
   WRITE (3,105) NAMESR
105 FORMAT (/5X,A5,29H SUBROUTINE NOT IN PRINT LIST//)
   CALL HED
   GO TO 83
21  ITNO(IST)=KT
   IGO = 2
   GO TO 200
22  GO TO 83
C  PRINT ITERATION NUMBERS OF SUBROUTINES
C
200 CONTINUE
   IPMAX = ISNB-1
   IF (IPMAX.EQ.0) WRITE (3,102) INSUB(1),ITNO(1)
102  FORMAT (/9X,4HSUB.,2X,A5,7H, ITER.,2X,I3)
   IF (IPMAX.EQ.1) WRITE (3,100) INSUB(1),ITNO(1),INSUB(2),ITNO(2)
100  FORMAT (/9X,4HSUB.,2X,A5,7H, ITER.,2X,I3,10X,4HSUB.,2X,A5,
1  7H, ITER.,2X,I3)
   IF (IPMAX.GE.2) GO TO 104
   GO TO 110
104 CONTINUE
   DO 201 IS=2,IPMAX
     WRITE (3,101) INSUB(IS),ITNO(IS),INSUB(IS+1),ITNO(IS+1)
101  FORMAT (15X,A5,9X,I3,16X,A5,9X,I3)
201 CONTINUE

```

```

110 CONTINUE
    GO TO (55,22),IGO
C
C CALL ENTRY
80 CONTINUE
    IF(MODE.EQ.3) GO TO 11
    IF (MHS) GO TO 11
    IF (VPR) GO TO 11
    CALL SKIP1
    CALL INP6(M,dolIZ,dolIZ,1,2,dolIZ)
    CALL SKIP1
    IF(MODE.EQ.2) GO TO 83
    IF (XSW) GO TO 83
C MODE IS 1. BEGINNING OF SUBROUTINE LOOP OR LOOPS
C ZERO ARRAYS FOR STORAGE OF NAMES AND ITERATION NUMBERS.
81 CONTINUE
    DO 12 KS=1,10
        INSUB(KS)=0
        ITNO(KS)=0
12 CONTINUE
    ISNB=0
11 NAMESR = BDAT(M,1)
    MSUBR = ELNUM(NAMESR)
    IF (MHS) GO TO 55
    IF (VPR) GO TO 55
    ISNB = ISNB+1
    INSUB(ISNB)=NAMESR
    ITNO(ISNB)=1
82 CONTINUE
    IF (MODE.NE.3) GO TO 55
    IGO = 1
    GO TO 200
55 CALL INFW(M,9,MSUBR)
C SETS SWITCH FOR INCREMENT TO RESTORE ORIGINAL VALUE
    CALL REPINT(MSUBR,2,0)
C SET BEGINNING OF COUNT FOR ITERATION NUMBER PRINT
    CALL REPINT(MSUBR,3,1)
C SET TOTAL COUNT FOR ITERATIONS THROUGH THE SUBROUTINE
    IF (KA.LT.1) KA = 1
    CALL REPINT(MSUBR,4,KA)
    IF (KA.EQ.1) CALL REPINT(MSUBR,2,1)
    STOR=.FALSE.
    RSRV=.FALSE.
    TRASW=.TRUE.
    TRA=MSUBR-1
C PUT MODE INTO IX OF INFO
    IF (.NOT.MSSW) CALL INFW(MODE,19,M)
C SET SWITCH FOR INCREMENT TO STORE ORIGINAL VALUE
    CALL REPINT(MSUBR,1,1)
    MODE=3
83 CONTINUE
    RETURN
    END

```

```

SUBROUTINE SUM(K,S,SM)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION S(1)
SM=0.
DO 10 I=1,K
  SM = SM + S(I)
10 CONTINUE
RETURN
END

C***** SW1 *****
SUBROUTINE SW1
IMPLICIT REAL*8 (A-H,O-Z)
C PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.
C PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.
C
  DIMENSION INF(24)
  INCLUDE 'INSTR.CCC'
  INCLUDE 'BSTORE.CCC'
  INCLUDE 'BINFF.CCC'
  INCLUDE 'BWORK.CCC'
  INCLUDE 'STORE.CCC'
  * LMAX AND MAX ARE DIMENSIONS OF STORE AND INFF

  INCLUDE 'CONTRL.CCC'
  INCLUDE 'DIM.CCC'
  INCLUDE 'BMI.CCC'
  INCLUDE 'BMI1.CCC'
  INCLUDE 'SWTCH.CCC'

  INCLUDE 'GRR.CCC'
  INCLUDE 'TRKINT.CCC'
  INCLUDE 'PLTIN.CCC'
  INCLUDE 'XXYY.CCC'
  INCLUDE 'BPLTCOM.CCC'
  INCLUDE 'SVNAM.CCC'
  INCLUDE 'COPY.CCC'
  common/seed/nr

  INTEGER BDAT,PVECS
  DIMENSION PAR(100),KREM(100),MATNM(20)
  INTEGER R7,ROT,PRD
  EQUIVALENCE (OPNAME,INF(1)),(PAR,KREM)
  DIMENSION BX(20)
  LOGICAL TABSW
  DATA IBLNK/4H /
  DATA CPYSW/.TRUE./
  DATA MINZER/-32767/
  DATA ISCL/9/
  DATA IBLNK,MXPMY,PVECS,R7,ROT,PRD/1H ,1,2,5,6,8/
  DATA (MATNM(I),I=1,12)/2HF ,2HD ,2HS ,2HT ,2HE ,2HK ,2HIA,2HIB,2HB
  1 ,2HA ,2HWD,2HWF/
C
  * IF OPNAME = NULL, IT IS A P OR C CARD. NEEDS SPECIAL
  * HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION

```

```

*      GO TO STOP, FIN, RUN, SUB, END, CALL, INCR, REPL, MESH, VPAR
*      =, CRD, BML, DRF, MAG, MAGV, EQU, INV, SHF, REF
*      **, MM, CYA, CYB, CYC, CELL, ALTC, TAB, PTAB, FITQ
C      FITB, FITR, FITV, PAGE, REM, WMA, SIZE, WBE, FIT.COPY, NCOPY, BEST, BEP
C      , BETA, KICK, RAND, SHF7, SOL, VAR, PRNT

```

```

      GO TO (1,2,3,4,5,6,19,20,74,74,
1      27,7,86,10,13,42,11,12,16,18,
2      21,14,23,24,23,29,30,81,82,25,
3      25,25,25,34,35,15,61,26,31,32,
4      33,23,43,44,45,46,47,48,49,50) , OP

```

```

*      STOP ENTRY
1  STOP
*      FIN ENTRY
2  CALL SRUN(2)
   RETURN
*      RUN ENTRY
3  CALL SRUN(1)
   RETURN
*      SUB ENTRY
4  CALL SUBR(1)
   RETURN
*      END ENTRY
5  CALL SUBR(2)
   RETURN
*      CALL ENTRY
6  CALL SUBR(3)
   RETURN
*      CRD ENTRY
7  CALL INP6(M,MXPMY,MINZER,
   . 12,1,0)
   IF (MODE.EQ.2) RETURN
   IF (XSW) RETURN
   CALL DATA (M,1,1,12,PAR)
   CALL RXRY(M,PAR)
   RETURN
C
C  DRIFT ENTRY
C  PROCESSES DRIFT DATA.  SETS UP DRIFT MATRICES IF IN EXEC.MODE.
10 IF ( MODE.EQ.3 ) GO TO 101
   IF (KB.EQ.0) CALL INP6(M,MXPMY,MINZER,1,1,0)
   IF (KB.GT.0) CALL INP8(M,MXPMY,MINZER,2,1,KB,2,0)
   IF (KA.EQ.0) GO TO 102
*      SET UP MATRIX SPACE FOR KA NUMBER OF DRIFTS.  WHOSE LENGTHS WILL
*  BE ELEMENTS OF INPUT VECTOR.
   CALL RSVMAT
   MIN = MIN - KA
102 IF (XSW) RETURN
   IF (ERROR) RETURN
101 IF (MODE.EQ.2) RETURN
   IF (KA.GT.0) GO TO 103
   EL = FDAT(M,1)

```



```

      IF (KB.EQ.0) GO TO 106
      FLSEP = FDAT(M,2)
      IF (EL.NE.0..AND.FLSEP.NE.0.) GO TO 106
C   VARIABLE DRIFT LENGTH
      FLTOT = 0.
      DO 104 K=1,KB
      MM = MDAT(M,K)
104  FLTOT = FLTOT + FDAT(MM,1)
      IF (FLSEP.NE.0 ) GO TO 105
      FLSEP = EL + FLTOT
      CALL REPFLT (M,2,FLSEP)
      GO TO 106
105  EL = FLSEP - FLTOT
      CALL REPFLT(M,1,EL)
106  CALL DRIFT(M,EL)
      RETURN
103  CALL DRFMAT(M,KA)
      RETURN
C
C   EQUATE ENTRY
C   PROCESSES EQU DATA. EQUATES MATRIX M TO N IF IN EXECUTION MODE.
C   MATRICES COULD HAVE VARIABLE DIMENSIONS.
11  IF (MODE.EQ.3) GO TO 1111
      CALL INP6(M,0,0,
      .1,2,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
1111 N=MDAT(M,1)
      CALL EQU(M,N)
      RETURN
C
C   INVERT ENTRY
C   PROCESSES INVERT DATA. INVERTS IF IN EXECUTION MODE.
12  IF (MODE.EQ.3) GO TO 1222
      KN = IBLNK
      IF (MSIZE.EQ.7) KN=R7
      IF (MSIZE.EQ.3) KN=MXPMY
      CALL INP6(M,KN,MINZER,
      . 1,2,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
1222 N=MDAT(M,1)
C   INVERT MATRIX AT INDEX N AND STORE IN INDEX M
      CALL INV(M,N)
      RETURN
C
C   MAGNET ENTRY
C   PROCESSES MAG DATA. SETS UP MAGNET MATRICES IF IN EXEC. MODE.
13  IF ( MODE.EQ.3 ) GO TO 131
      CALL INP6(M,MXPMY,MINZER,6,1,0)
      IF (KA.EQ.0) GO TO 132
*       SET UP MATRIX SPACE FOR KA NUMBER OF MAGNETS.  SOME OF THE
*   INPUT PARAMETERS DERIVE FROM PREVIOUSLY DEFINED VECTOR.
      CALL RSVMAT
      MIN = MIN - KA

```

```

132 IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
    IF (ERROR) RETURN
131 IF (KA.GT.0) GO TO 133
    CALL DATA (M,1,1,6,PAR)
    CALL MAGNET(M,PAR)
    RETURN
133 CALL MAGMAT(M,KA)
    RETURN

```

C

C KICK ENTRY

```

45 IF (MODE.EQ.3) GO TO 451
    CALL INP8(M,R7,MINZER,1,2,3,1,0)
    IF ( (MODE.EQ.2).OR.(XSW).OR.(ERROR) ) RETURN
451 CALL KICK(M)
    RETURN

```

C

C ~~MM~~ ENTRY

C PROCESSES ~~MM~~ DATA. MULTIPLIES MATRICES IF IN EXEC. MODE

```

14 IF ( MODE.EQ.3 ) GO TO 141
    IF (MSIZE.NE.3) KN = R7
    CALL INP6(M,KN,MINZER,1,-2,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
141 NDAT=INFF(17,M)
    CALL MIFILL(M,1,NDAT,NELS,MI)
    KB = INFF(5,M)
    IF (KB.NE.0.AND.KB.NE.1) GO TO 142
    CALL MM(M,NELS,MI)
    RETURN
142 CALL EXECMM(M,NELS,KB,MI)
    RETURN

```

C

C ~~WMA~~ ENTRY

C PROCESSES ~~WMA~~ DATA. WRITES MATRICES IF IN EXEC.MODE.

```

15 IF (MODE.EQ.3) GO TO 1555
    CALL INP6(M,0,0,
      KA,2,0)
    IF(MODE.EQ.2) RETURN
    IF (XSW) RETURN
1555 CALL MLIST(M,1,KA,MI)
    IF(ERROR) RETURN
    CALL WMA(KA)
    RETURN

```

C

C SHIFT ENTRY

C PROCESSES SHF DATA. SETS UP SHIFT MATRIX IF IN EXEC. MODE

```

16 IF (MODE.EQ.3) GO TO 1666
    CALL INP6(M,MXPMY,MINZER,
      4,1,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
1666 CALL DATA(M,1,1,4,PAR)
    CALL SHF(M,PAR)

```

```

      RETURN
C
C REFLECT ENTRY
C PROCESSES REF DATA. COMPUTES MATRIX REFLECTION IF IN EXEC.MODE
18  IF (MODE.EQ.3) GO TO 1888
      KN = IBLNK
      IF (MSIZE.EQ.7) KN = R7
      IF (MSIZE.EQ.3) KN = MXPMY
      CALL INP6(M,KN,MINZER,
        . 1,2,0)
      IF (MODE.EQ.2) RETURN
      IF (.XSW) RETURN
1888  N=MDAT(M,1)
      CALL REF(M,N)
      RETURN
C
C INCR ENTRY
*   DETERMINE TYPE TO BE INCREMENTED— FL.PT., INTEGER, KA OR KB.
19  IF (MODE.EQ.3) GO TO 192
      DECODE (8,201,ICARD(7)) KT
      KE = 0
      IF (KT.EQ.1H ) KE = 1
      IF (KT.EQ.1HF) KE = 1
      IF (KT.EQ.1HI) KE = 3
      IF (KT.EQ.2HKA) KE = 3
      IF (KT.EQ.2HKB) KE = 3
      IF (KE.NE.0) GO TO 191
*   ILLEGITIMATE TYPE
      WRITE (3,190) KT
190  FORMAT(5X,1H*, 'ERROR',2H* ,A5,' IS AN ILLEGITIMATE TYPE OF INCREME
      INT. ')
      ERROR = .TRUE.
      RETURN
191  CONTINUE
      CALL RESRV8(M,0,0,
        . 2,2,2,KE,0)
      CALL LOAD6(M,
        . 2,2,1,KE,0)
      IF (MODE.EQ.2) GO TO 193
192  IF (.XSW) RETURN
      KO = KA
      CALL INCR(M,KO)
      RETURN
193  CALL INFW(MSUBR,19,M)
      RETURN
C
C REPL ENTRY
20  CONTINUE
      IF (MODE.EQ.3) GO TO 204
*   DETERMINE TYPE OF REPLACEMENT — FL. PT., INTEGER, BCD OR KA,KB
      DECODE (8,201,ICARD(7)) KT
201  FORMAT (1X,A4,3X)
      KE = 0
*   IF BLANK, DEFAULT IS F
      IF (KT.EQ.1H ) KE =1

```

```

      IF (KT.EQ.1HF) KE = 1
      IF (KT.EQ.2HSF) KE = 1
      IF (KT.EQ.1HQ) KE = 1
      IF (KT.EQ.1HH) KE = 2
      IF (KT.EQ.1HI) KE = 3
      IF (KT.EQ.2HKA) KE = 3
      IF (KT.EQ.2HKB) KE = 3
      IF (KE.NE.0) GO TO 202
*      ILLEGITIMATE TYPE
      WRITE (3,203) KT
203  FORMAT (5X,1H*, 'ERROR', 2H* ,A5, ' IS AN ILLEGITIMATE TYPE OF REPLAC
      1EMENT. ')
      ERROR = .TRUE.
      RETURN
202  CONTINUE
      CALL INP8(M,0,0,
      . 2,2,1,KE,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
204  CONTINUE
      CALL REPL(M)

      RETURN
C
C  STAR ENTRY
C  PROCESSES STAR DATA. COMPUTES IF IN EXEC. MODE
21  IF (MODE.EQ.3) GO TO 2111
      KN = IBLNK
      IF (MSIZE.EQ.7) KN=R7
      IF (MSIZE.EQ.3) KN=MXPMY
      CALL INP6(M,KN,MINZER,
      . 1,2,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
2111 CALL STAR(M,KA)
      RETURN
C
C
C  CYA AND CYC ENTRY
*      CYA, CYC, BEST ENTRY
23  IF (MODE.EQ.3) GO TO 543
C  MAKE RESERVATION AND STORE INPUT DATA
      KN = MXPMY
      CALL RESRVA(M,KN,MINZER,
      . 4,2,2,1,2,3,0)
      CALL LOADA(M,4,2,2,1,1,4,2,3,0)
      NDAT = 1
      IF (KN.EQ.IBLNK) KN = INFF(20,M)
      IF (KN.EQ.2HR7) KN = R7
      IF (KN.EQ.3HMX) KN = MXPMY
      IF (OPNAME.NE.3HCYA.AND.KA.NE.21) GO TO 521
      IF (KA.GT.999) GO TO 545
      DECODE (1,520,NAME) NM
520  FORMAT (A1)
      CALL MIFILL(M,1,NDAT,NELS1,MI1)

```

```

      IF ( OPNAME.EQ.3HCYA) GO TO 522
      NELS2=NELS1
      IF (KB.LT.0) NELS2=NELS1+1
      NQ3 = 14*(NELS2+1) + 18
      LQ3 = LFILE + 1
      LFILE = LQ3 + NQ3
      NTOT = NTOT + NQ3
      INFF(18,M) = NTOT
      INFF(24,M) = LQ3
      IF (MODE.EQ.2) GO TO 542
      IF (XSW) RETURN
      GO TO 544
522  CALL DEFSET(M,NM,NELS1,MATNM,0,KN)
521  IF (MODE.EQ.2) GO TO 542
      IF (XSW) RETURN
543  NDAT = 1
      CALL MIFILL(M,1,NDAT,NELS1,MI1)
544  CALL CYX(M)
      IF (MODE.EQ.3) RETURN
542  IF (OPNAME.NE.4HCYA ) RETURN
      MIN = MIN - NELS1
      RETURN
545  WRITE (3,546)
546  FORMAT (23H CYA COUNT EXCEEDS 999.)
      ERROR=.TRUE.
      RETURN
C
C  CYB  ENTRY
24   IF (MODE.EQ.3) GO TO 240
      KN = MXPMY
      CALL INP8(M,0,0,
        .1,2,1,3,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
C  RETRIEVE ELEMENT NAME FROM DATA AND OBTAIN ITS INDEX.
240  CONTINUE
      CALL SETCYB(M,KA)
      IF (ERROR) RETURN
245  CALL CYX(M)
      RETURN
C
C          FITQ,FITB,FITR,FITY ENTRY
25   CALL SETFIT
      RETURN
C
C  WBE ENTRY
C  PROCESSES WBE DATA. CALLS WBE TO WRITE BETATRON FUNCTIONS
C    IF IN EXECUTION MODE
26   IF (MODE.EQ.3) GO TO 2666
      CALL INP6(M,0,0,
        . KA,2,0)
      IF(MODE.EQ.2) RETURN
      IF (XSW) RETURN
2666  CALL MLIST(M,1,KA,MI)

```

```

      IF(ERROR) RETURN
      CALL WBE(KA,KB)
      RETURN

```

C

C = ENTRY

```

27  IF (MODE.EQ.3) GO TO 275
    KN = IBLNK
    DECODE (8,5000,ICARD(8)) IOP
5000 FORMAT (2X,A4,2X)
    IF (IOP.NE.IBLNK) GO TO 271

```

C KIND AND TYPE ARE SCALAR AND SINGLE

```

    KN = ISCL
    CALL INPB(M,KN,0,
. 1,1,1,2,0)
    RETURN

```

C LOOK AT FIRST INPUT PARAMETER

```

271 CALL CHKD(ICARD(6),KN,ITP)
    IF (KN.NE.4HSCAL) GO TO 272
    IQ = 1
    KN = ISCL
    IF (ITP.NE.4HSNGL) IQ = 2
    CALL INPA(M,KN,IQ,
. 1,1,1,2,2,1,0)
    F = FDAT(M,3)
    CALL REPLQ(M,1,1,F)
    IF (MODE.EQ.2.OR.XSW) RETURN
    GO TO 276

```

C MATRIX. INPUT IS BCD,BUT IN 10 COLUMN FIELDS.

```

272 IF (KN.EQ.3HMXV) KN=1
    IF (KN.EQ.2HR7) KN = 5
    CALL INPA(N,KN,MINZER,
. 1,2,1,2,1,2,0)
    IF (MODE.EQ.2.OR.XSW) RETURN
    GO TO 277
275 IF (XSW) RETURN
    KN = INFF(20,M)
    IF (KN.NE.4HSCAL) GO TO 277
    IOP = BDAT(M,1)

```

C RETRIEVE OPERATION SYMBOL. IF KIND NOT SCALAR,IT IS 2ND BCD INPUT

C IF SINGLE SYMBOLIC ARGUMENT, GET ITS VALUE

```

    IF (IOP.EQ.IBLNK) RETURN
    ITP = INFF(21,M)
276 IQ = 2
    IF (ITP.NE.4HSNGL) IQ = 4
    CALL DATA (M,1,1,IQ,PAR)
    CALL AROP(M,PAR,IOP)
    RETURN

```

```

277 KREM(1) = BDAT(M,1)
    KREM(2) = BDAT(M,3)
    IOP = BDAT(M,2)
    CALL AROP(M,KREM,IOP)
    RETURN

```

C

C CELL ENTRY

```

29  IF(MODE.EQ.3) GO TO 293
C  MAKE RESERVATIONS FOR FIRST CYCLED MATRIX AND STORE INPUT DATA
C  RESERVE SPACE IN DATA FOR THE MI LIST
      CALL RESRV8 (M,MXPMY,MINZER,
        . 7,1,12,3,0)
      CALL LOAD4(M,
        . 7,1,0)
C  SET UP SPACE FOR CYCLED MATRICES
      DECODE (1,520,NAME) NM
      CALL DEFSET (M,NM,12,MATNM,4,MXPMY)
      IF (MODE.EQ.2) GO TO 294
      IF (XSW) RETURN
293  CALL DATA(M,1,1,7,PAR)
      CALL CELL(M,PAR)
      IF (MODE.EQ.3) RETURN
294  MIN=MIN-16
      RETURN
C
C  ALTC ENTRY
30  IF (MODE.EQ.3) GO TO 301
C  MAKE RESERVATIONS FOR FIRST CYCLED MATRIX AND STORE INPUT
C  RESERVE SPACE IN DATA FOR MI LIST
      CALL RESRV8(M,MXPMY,MINZER,
        . 8,1,12,3,0)
      CALL LOAD4(M,
        . 8,1,0)
      DECODE (1,520,NAME) NM
C  SET UP SPACE FOR CYCLED MATRICES AND QF,QD,QS,QT
      CALL DEFSET (M,NM,12,MATNM,4,MXPMY)
      IF (MODE.EQ.2) GO TO 302
      IF (XSW) RETURN
301  CALL DATA(M,1,1,8,PAR)
      CALL ALTC(M,KA,PAR)
      IF(MODE.EQ.3) RETURN
302  MIN=MIN-16
      RETURN
C
*          FIT ENTRY
31  IF (MODE.EQ.3) GO TO 311
      K1 = KA*2
      K2 = KB*2
      K = K1 + K2 + 2
      CALL RESRV8(M,0,0,
        . K,2,1,3,0)
      CALL LOAD6(M,
        . 2,2,1,3,0)
      CALL LOAD4(M,
        . K1,2,0)
      CALL LOAD4(M,
        . K2,2,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
311  CONTINUE
c    CALL DOFIT(M)
      RETURN

```

```

C          COPY ENTRY
32  IF (MODE.EQ.3) GO TO 320
    CALL INP4(M,0,0,0)
320 IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
    CPYSW = .TRUE.
    RETURN

C          NCPY ENTRY
33  IF (MODE.EQ.3) GO TO 330
    CPYSW = .FALSE.
    CALL INP4(M,0,0,0)
    WRITE(3,345)
345 FORMAT(/2X,'***',3X,'...',4X,'...',9X,'//',2X,20(' ')/)
330 IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
    RETURN

C PAGE ENTRY
C RESTORES PAGE IF IN EXECUTION MODE
34  IF (MODE.EQ.3) GO TO 341
*      STORE DATA FIELD TO PRINT AS A COMMENT
    CALL INP6(M,0,0,
      15,2,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
341 K0 = INFF(17,M)
    CALL DATA (M,2,1,K0,KREM)
    IF (CPYSW) WRITE (3,342) (KREM(I),I=1,K0)
342 FORMAT (1H1/8X,20A4)
    RETURN

C
C REM ENTRY
C PRINTS REMARKS
35  IF (MODE.EQ.3) GO TO 352
    CALL INP6(M,0,0,
      15,2,0)
352 IF (XSW) RETURN
    K0 = INFF(17,M)
    CALL DATA (M,2,1,K0,KREM)
    IF (CPYSW) WRITE (3,353) (KREM(I),I=1,K0)
353 FORMAT (/8X,20A4)
    RETURN

C
* MAGV ENTRY
42  CONTINUE
    IF (MODE.EQ.3) GO TO 421
    CALL INP6(M,R7,MINZER,
      6,1,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
421 CALL DATA (M,1,1,6,PAR)
    CALL MAGV(M,PAR)
    RETURN

```



```

43  IF (MODE.EQ.3) GO TO 431
C   INITIALIZE FOR PLOT PACKAGE ONCE ONLY
432  CALL INPA(M,0,0,
      . 2,2,2,3,3,1,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
431  CALL PLTBET(M)
      RETURN
C
C   SIZE ENTRY, SWITCH DEFINING MATRIX SIZE
61  IF (MODE.EQ.3) RETURN
C   SET MSZ TO KA. IF KA = 7 FOR 7X7 , 3 FOR 3X3, 37 IF NOT KNOWN
      MSIZE = KA
      IF (CPYSW) WRITE (3,610) NAME,OPNAME,KA,KB
610  FORMAT (6H *** ,1X,A5,2X,A5,1X,I3,1X,I3)
      RETURN
*
      MESH AND VPAR ENTRY
74  CONTINUE
      IF (MODE.EQ.3) GO TO 741
      KL = KA + 1
      KM = 3*KA
      CALL RESRVA(M,0,0,
      . KL,2,KA,3,KM,1,0)
      CALL LOAD4(M,
      . 1,2,0)
      DO 740 IJ=1,KA
      CALL LOAD8(M,
      . 1,3,1,2,3,1,0)
740  CONTINUE
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
741  CONTINUE
      CALL MESH(M,KA,OPNAME)
      RETURN
C   TAB ENTRY
81  IF (MODE.EQ.3) GO TO 810
      IF (KA.GT.10) KA = 10
      KS = KA*60+1
      CALL INP6(M,0,KS,
      . KA,2,0)
C   ZERO ARRAY AT LQ
      DO 811 I=1,60
811  PAR(I) = 0.
      IK = 2
      DO 812 I=1,KA
      CALL STDAT(M,5,IK,60,PAR)
      IK = IK + 60
812  CONTINUE
C   SET CURSOR = 2 AT LQ
      CALL REPLQ(M,1,1,2)
      TABSW = .FALSE.
C   SET UP INDICES TABLE
      CALL TABULT(M,TABSW)
      IF (MODE.EQ.2) RETURN
810  TABSW = .TRUE.

```

```

      CALL TABULT(M,TABSW)
      RETURN
C PTAB ENTRY
82  IF (MODE.EQ.3) GO TO 820
      CALL INP6(M,0,0,
        . KA,2,0)
      IF (MODE.EQ.2) RETURN
820  CALL PRNTAB(M)
      RETURN
*          BML ENTRY — BEAM LINE
86  IF (MODE.EQ.3) RETURN
      CALL INP6(M,0,0,
        . 1,-2,0)
      RETURN

C BETA ENTRY
44  IF(MODE.EQ.3) GO TO 441
      CALL INP8(M,ISCL,MINZER,
        . 1,2,1,1,0)
      IF(MODE.EQ.2) RETURN
441  MMMDAT(M,1)
      CALL BET(MM,BX(1),BX(11),0)
      X=BX(KA)
      CALL REPFLT(M,0,X)
      RETURN

C RAND ENTRY
C
46  IF(MODE.NE.3) CALL INP6(M,ISCL,0,3,1,0)
      IF(MODE.EQ.2) RETURN
      RNS=FDAT(M,3)
      NRS=RNS
c If third field non-zero, use it as random seed
      IF(RNS.GT.0.) nr=nrs
      KB=MAX0(KB,1)
      IF(KA.LE.0) RETURN
C RUN RANDOM GENERATOR KA*KB TIMES
      KB = MAX0(KB,1)
      DO461 I=1,KA
      DO461 J=1,KB
461  X=RAN(nr)
      RETURN

C SHF7 ENTRY 7X7 SHIFT MATRIX
47  IF (MODE.EQ.3) GO TO 471
      CALL INP6 (M,R7,MINZER,
        . 6,1,0)
      IF (MODE.EQ.2) RETURN
471  CALL DATA (M,1,1,6,PAR)
      CALL SHF7(M,PAR)
      RETURN

C SOLENOID ENTRY
48  IF (MODE.EQ.3) GO TO 481
      CALL INP6 (M,R7,MINZER,
        . 4,1,0)

```

```

      IF (MODE.EQ.2) RETURN
481  CALL DATA (M,1,1,4,PAR)
      CALL SOL(M,PAR)
      RETURN
C      VAR ENTRY
C      DEFINES VARIABLE BY INSTRUCTION NAME AND POSITION.
C.....
C  VNAM  VAR  KA  NAME
C.....
C  EXAMPLE 1
C    QF  MAG          1.5      0.06      1.
C    GQF VAR    1    QF
C    QFH MAG          0.75      GQF      1
C.....
C.....
C
C  IN THIS EXAMPLE QFH HAS HALF THE LENGTH OF QF, BUT THE SAME
C  GRADIENT, 0.06.
C
C  EXAMPLE 2 — A MORE USEFUL EXAMPLE
C    L    DRF          2.
C    SR    SUB
C    QF  MAG          1.      0.05      1.
C    QD  MAG          1.      -0.05      1.
C    C    MM      QF  L  QD  L
C    END
C    FITQ          SR  C  QF  QD          2      2      .25  .25
C
C    GF  VAR    2    QF
C    GD  VAR    1    QD
C    QFH MAG          .75      GF          1.
C    QDH MAG          .75      GD          1.
C    CYC          -1 QFH  L  QDH
C.....
C
49  IF (MODE.EQ.3) GO TO 491
      CALL INP6 (M,0,0,1,2,0)
      IF (MODE.EQ.2) RETURN
491  FVAR = FLDAT(M,1)
      NAM=MNAME(M)
      WRITE (3,490) NAM,FVAR
490  FORMAT (30X,A5,' VALUE — ',E15.8)
C
      RETURN
C  PRNT ENTRY — PRINT SELECTED DATA
50  IF (MODE.EQ.3) GO TO 51
      CALL INP6(M,0,0,1,-2,0)
      IF (MODE.EQ.2) RETURN
51  IF (XSW) RETURN
      CALL PRNT(M)
      RETURN

      END
C ***** SW3 *****
      SUBROUTINE SW3

```

IMPLICIT REAL*8 (A-H,O-Z)

C PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.
C PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.
C

DIMENSION INF(24)
LOGICAL FSW
INCLUDE 'INSTR.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'STORE.CCC'
INCLUDE 'CONTRL.CCC'
INCLUDE 'DIM.CCC'
INCLUDE 'BMI.CCC'
INCLUDE 'SWTCH.CCC'
INCLUDE 'GRR.CCC'
INCLUDE 'COPY.CCC'

LOGICAL*1 ISGN,BLNK/' '/,DASH/'-'/
INTEGER BDAT,PVECS,R7,ROT,PRD
DIMENSION PAR(100),KREM(100),NCELL(5),JMAG(7)
DATA JMAG /2HIF,2HID,2HOF,2HOD,1HE,1HS,1HT/
DATA MXPMY,PVECS,R7,ROT,PRD/1,2,5,6,8/
EQUIVALENCE (OPNAME,INF(1)),(PAR(1),KREM(1))
DIMENSION MATNM(20),MXLIST(20)
DATA MXPMY/1/
DATA MINZER/-32767/
DATA MATNM/2HF,2HD,2HS,2HT,2HE,2HK,2HIA,2HIB,2HB,
1 2HA,2HWD,2HWF,8*' '/

C DATA MATNM/'F','D','S','T','E','K','IA','IB','HB'
C 1,'A','WD','WF',8*' '/
C

C IF OPNAME = NULL, IT IS A P OR C CARD. NEEDS SPECIAL
C HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION

NOP = OP - 50

GO TO (36,80,80,49,22,17,28,37,38,39,
1 47,44,48,46), NOP

C GO TO SUM, ACT, DELE, SCOP, TEST, WFL, MAGS, CELC, CFD, DCFD
C STR2,STR4,STRP,STRN

17 IF (MODE.EQ.3) GO TO 171
CALL INP6(M,0,0,1,-2,0)
IF (MODE.EQ.2) RETURN
171 OP = INFF(3,M)
IF (OP.LT.0) RETURN
KA = INFF(4,M)
IF (KA.EQ.0) RETURN
CALL WFLSR(M)
RETURN

C
22 CALL TEST(M)
RETURN

C
C

```

C  MAGS ENTRY
C  PROCESSES MAGS DATA.CALCULATES TRANSFER MATRIX FOR NEW MAGNET IF IN
C  EXECUTION MODE
28  IF ( MODE.EQ.3 ) GO TO 281
    CALL INP6(M,MXPMY,MINZER,1,2,5,1,0)
    IF(MODE.EQ.2) RETURN
    IF (XSW) RETURN
281  CALL DATA (M,1,1,5,PAR)
    N=MDAT(M,1)
    CALL MAGS(M,N,PAR)
    RETURN
C  SUM ENTRY
36  IF(MODE.EQ.3) GO TO 361
    CALL INP6(M,0,1,KA,1,0)
    IF(MODE.EQ.2) RETURN
    IF (XSW) RETURN
361  CALL DATA (M,1,1,KA,PAR)
    CALL SUM(KA,PAR,RES)
    CALL STDA7(M,5,1,1,RES)
    RETURN
C
C  CELC ENTRY
37  IF (MODE.EQ.3) GO TO 371
C  MAKE RESERVATIONS FOR FIRST CYCLED MATRIX AND STORE INPUT DATA
C  RESERVE SPACE IN DATA FOR THE MI LIST
    CALL RESRV8(M,MXPMY,MINZER,8,1,12,3,0)
    CALL LOAD4(M,7,1,0)
    DECODE(1,520,NAME)NM
C  SET UP SPACE FOR CYCLED MATRICES AND QF,QD,QS,QT,QE
    CALL DEFSET(M,NM,12,MATNM,5,MXPMY)
    IF (MODE.EQ.2) GO TO 372
    IF (XSW) RETURN
371  CALL DATA (M,1,1,8,PAR)
C  CALL CELC(M,PAR)
    IF(MODE.EQ.3) RETURN
372  MIN=MIN-17
    RETURN
C
C  CFD ENTRY
38  IF (MODE.EQ.3) GO TO 381
C  MAKE RESERVATIONS FOR FIRST CYCLED MATRIX AND STORE INPUT DATA
C  RESERVE SPACE IN DATA FOR THE MILIST
    CALL RESRV8(M,MXPMY,MINZER,8,1,12,3,0)
    CALL LOAD4(M,8,1,0)
    DECODE(1,520,NAME)NM
C  SET UP SPACE FOR CYCLED MATRICES AND QF,QD,QS,QT
    CALL DEFSET(M,NM,12,MATNM,4,MXPMY)
    IF (MODE.EQ.2) GO TO 382
    IF (XSW) RETURN
381  CALL DATA(M,1,1,8,PAR)
C  CALL CFD(M,PAR)
    IF (MODE.EQ.3) RETURN
382  MIN=MIN-16
    RETURN
C

```

C DCFD ENTRY

```

39  IF(MODE.EQ.3) GO TO 391
    CALL RESRV8(M,MXPMY,MINZER,9,1,12,3,0)
    CALL LOAD4(M,9,1,0)
    DECODE(1,520,NAME)NM

```

C SET UP SPACE FOR CYCLED MATRICES AND QF,QD,QS,QT

```

    CALL DEFSET(M,NM,12,MATNM,4,MXPMY)
    IF (MODE.EQ.2) GO TO 392
    IF (XSW) RETURN

```

```

391 CALL DATA(M,1,1,9,PAR)

```

C CALL DCFD(M,PAR)

```

    IF (MODE.EQ.3) RETURN

```

```

392 MIN = MIN-16

```

```

    RETURN

```

C STR4 ENTRY. PROCESSES STR4 DATA

```

44  IF (MODE.EQ.3) GO TO 442
    CALL INP8(M,MXPMY,MINZER,3,2,4,1,0)
    M1 = M - 1
    DO 441 I=1,4
    CALL MAGRSV (-M1,JMAG(I),0,0,NAME)

```

```

441 M1 = M1 - 1

```

```

    DO 440 I=5,7

```

```

    CALL DRFRSV(-M1,JMAG(I),0,0,NAME)

```

```

440 M1 = M1 - 1

```

```

    IF (MODE.EQ.2) GO TO 443

```

```

    IF (XSW) RETURN

```

```

442 CALL DATA(M,2,1,3,NCELL)

```

```

    CALL DATA(M,1,1,4,PAR)

```

```

    CALL STR4(M,NCELL,PAR)

```

```

    IF (MODE.EQ.3) RETURN

```

```

443 MIN = MIN - 7

```

```

    RETURN

```

C

C

C STRN ENTRY.

```

46  IF (MODE.EQ.3) GO TO 462
    IF (KA.NE.2) KA=1
    KF = 4
    IF(KA.EQ.2) KF=8
    CALL RESRV8(M,MXPMY,MINZER,2,2,KF,1,0)
    CALL LOAD6(M,2,2,4,1,0)
    IF (KA.EQ.2) CALL LOAD4(M,4,1,0)
    DECODE(1,520,NAME)NM
    DO 461 I = 1,6

```

```

461 MXLIST(I) = MATNM(I + 6)

```

```

    MXLIST(7)= MATNM(5)

```

```

    MXLIST(8) = MATNM(3)

```

```

    M1 = M - 1

```

```

    DO 460 I=1,6

```

```

    CALL MAGRSV(-M1,MXLIST(I),0,0,NAME)

```

```

460 M1 = M1 - 1

```

```

    DO 464 I=7,8

```

```

    CALL DRFRSV(-M1,MXLIST(I),0,0,NAME)

```

```

464 M1 = M1 - 1

```

```

    IF (MODE.EQ.2) GO TO 463

```

```

        IF (XSW) RETURN
462 CALL DATA(M,2,1,2,NCELL)
        KF = 4
        IF (KA.EQ.2) KF = 8
        CALL DATA(M,1,1,KF,PAR)
C      CALL STRN(M,NCELL,PAR)
        IF (MODE.EQ.3) RETURN
463 MIN = MIN - 8
        RETURN
C STR2 ENTRY. DESIGN COLLINS STRAIGHT SECTION FOR INSERTION.
47  IF (MODE.EQ.3) GO TO 502
        CALL RESRVA(M,MXPMY,MINZER,1,2,4,1,6,3,0)
        CALL LOAD6 (M,1,2,4,1,0)
        DO 501 I = 1,3
501  MXLIST(I) = MATNM(I)
        MXLIST(4) = MATNM(5)
        DECODE(1,520,NAME)NM
        CALL DEFSET(M,NM,0,MXLIST,4,MXPMY)
        M1 = BDAT(M,1)
        M2 = M - 5
        ENCODE(8,504,INFF(2,M2))M1,MATNM(6)
504  FORMAT (A1,A4,3X)
        CALL RESRV6(M-5,MXPMY,MINZER,13,3,0)
        IF (MODE.EQ.2) GO TO 503
        IF (XSW) RETURN
502 CALL STR2(M)
        IF (MODE.EQ.3) RETURN
503 MIN = MIN - 5
        RETURN
C STRPI ENTRY. DESIGN PI-TYPE STRAIGHT SECTION FOR INSERTION.
48  IF (MODE.EQ.3) GO TO 512
        CALL RESRVA(M,MXPMY,MINZER,1,2,5,1,10,3,0)
        CALL LOAD6(M,1,2,5,1,0)
        DECODE(1,520,NAME)NM
        CALL DEFSET(M,NM,0,MATNM,4,MXPMY)
        IF (MODE.EQ.2) GO TO 513
        IF (XSW) RETURN
512 CALL STRPI(M)
        IF (MODE.EQ.3) RETURN
513 MIN = MIN - 4
        RETURN
C SCOPE ENTRY
49  CALL INP4(M,0,0,0)
        IF (MODE.EQ.2) RETURN
        IF (XSW) RETURN
C      CALL SCOPCN
        RETURN
C ACT OR DELE ENTRY
80  CONTINUE
        IF (MODE.EQ.3) GO TO 800
C IF KB=0, INPUT CONTAINS INDEFINITE NO. OF ELEMENTS TO BE CHANGED.
C IF KB GT 0, THEN THE LARGER OF KA OR KB ELEMENTS WILL BE CHANGED.
C BEGINNING WITH THE ONE NAMED INPUT ELEMENT.
        IF (KB.NE.0) CALL INP6(M,0,0,1,2,0)
        IF (KB.EQ.0) CALL INP6(M,0,0,1,-2,0)

```

```

      IF (MODE.EQ.2) RETURN
      IF(XSW) RETURN
800  CALL XEQCON(M)
      RETURN
520  FORMAT(A1)
      END
C ***** SW4 *****
      SUBROUTINE SW4
      IMPLICIT REAL*8 (A-H,O-Z)
C  PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.
C  PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.
C
      DIMENSION INF(24)
      INCLUDE 'INSTR.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'STORE.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'DIM.CCC'
      INCLUDE 'BMI.CCC'
      INCLUDE 'SWTCH.CCC'
      INCLUDE 'GRR.CCC'
      INCLUDE 'MOVTOG.CCC'
      INCLUDE 'CONST.CCC'
      common /seed/nr

      INTEGER BDAT,PR
      DIMENSION T(49),RW(3),V0(7)
      DIMENSION PAR(100),KREM(100),NCELL(5),IMAG(7),JMAG(7)
      INTEGER R7,ROT,PRD
      DATA R7,ROT,MAP,PRD/5,6,7,8/
      EQUIVALENCE (OPNAME,INF(1)),(PAR(1),KREM(1))
      DATA V0/7*0./
      DATA MINZER/-32767/

C
C      IF OPNAME = NULL, IT IS A P OR C CARD.  NEEDS SPECIAL
C      HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION

      NOP = OP - 64
C      GO TO FXPT,TRK,PRD,SXTP,MAP,BMIS,EMIS,ROTZ,ROT,INV2,ELQ,MOVE,
C      MODE,CON,VAR,NPOL,ORBC
      GO TO (40,35,34,33,1000,8,9,41,42,43,44,10,10,39,45,50,55,56,57),
1      NOP

8      CALL BMIS
10     RETURN
9      CALL EMIS
      RETURN
C
C
C      SXTP ENTRY
33     IF (MODE.EQ.3) GO TO 331
      CALL INP6(M,MAP,MINZER,4,1,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
331    CALL DATA(M,1,1,4,PAR)

```



```

      IF (MODE.EQ.1) PAR(4) = 0.
      DO 332 I=1,6
332  V0(I)=0.
      CALL SXTP(M,V0,PAR)
      RETURN

C PRD ENTRY  DEFINE A PRODUCT OF LINEAR AND NON-LINEAR TRANSFORMATION
34  IF (MODE.EQ.3) RETURN
      CALL INP6(M,PRD,MINZER,KA,2,0)
      RETURN

C TRK ENTRY
35  IF (MODE.EQ.3) GO TO 352
      CALL RESRVA(M,R7,MINZER,4,2,6,3,4,1,0)
      CALL LOAD6(M,4,2,6,3,0)
      CALL LOAD4(M,4,1,0)
      IF(MODE.EQ.2) RETURN
      IF (XSW) RETURN
352  CALL TRK(M)
      RETURN

C FXPT ENTRY
40  IF (MODE.EQ.3) GO TO 401
C  RESERVE SETS UP SPECIAL LQ LOCATIONS FOR FXPT
      CALL RESRVA(M,R7,0,5,1,2,2,5,3,0)
      CALL LOAD6(M,2,2,5,3,0)
      CALL LOAD4(M,5,1,0)
      IF(MODE.EQ.2) GO TO 402
      IF (XSW) RETURN
401  CALL FXPT(M)
402  RETURN

C ROTZ ENTRY
41  IF(MODE.EQ.3) RETURN
      CALL INP6(M,ROT,MINZER,1,1,0)
      RETURN

C
C ROT ENTRY——ROTATE MATRIX
42  IF (MODE.EQ.3) GO TO 420
      CALL INP8(M,R7,MINZER,1,2,1,1,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
420  MB = MDAT(M,1)
      THETA = FDAT(M,1)
      THETA = THETA*RADEG
      CALL ROTM(M,MB,THETA)
      RETURN

C
      INV2 ENTRY  ROTATE MATRIX 180 DEGREES AND REFLECT
43  IF (MODE.EQ.3) GO TO 430
      CALL INP6(M,R7,MINZER,1,2,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
430  MB = MDAT(M,1)
      CALL INV2(M,MB)
      RETURN

```

C ELQ ENTRY

```

44 IF (MODE.EQ.3) RETURN
   CALL INP8(M,MAP,MINZER,1,2,5,1,0)
   RETURN

```

C MAP ENTRY

```

1000 IF (MODE.EQ.3) RETURN
     CALL INP6(M,MAP,MINZER,KA,1,0)
     RETURN

```

C MOVE ENTRY

```

39  IF (MODE.EQ.3) GO TO 390
    CALL RESRV8(M,R7,MINZER,3,2,10,1,0)
    CALL LOAD6(M,3,2,4,1,0)
    MQ = MDAT(M,1)
    NOPQ = INFF(1,MQ)
    IF (NOPQ.EQ.4HSXTP) INFF(20,M) = 3HMAP
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
390 DO 391 I=1,7
391 V0(I)=0.
    nrrn=nrr
    CALL MOVE(M,M,V0)
    TOG = .TRUE.
    nrr=nrrn
    RETURN

```

SW4.185
SW4.186
SW4.187
SW4.188
MAY0SW4.
MAY0SW4.
MAY0SW4.
SW4.189
SW4.190
SW4.191
SW4.192
SW4.193
SW4.194
SW4.195

C MOD ENTRY

```

C
C   THIS INSTRUCTION MAY BE USED TO SEE EXPLICITLY THE EFFECT OF
C   SUBROUTINE REV MAT. WHICH MAKES A NEW MATRIX WHICH IS THE TRANSFER
C   MATRIX FOR RAYS IN THE NEIGHBORHOOD OF A REFERENCE TRAJECTORY.
C   EXAMPLE
C
C   V   PVEC      .005      0.      0.      0.      0.      0.
C   QF  MAG      1.25      .075     1.
C   QFV MOD      QF      V
C

```

SW4.196
SW4.197
SW4.198
SW4.199
SW4.200
SW4.201
SW4.202
SW4.203
SW4.204
SW4.205
SW4.206

```

45  IF (MODE.EQ.3) GO TO 450
    CALL INP6(M,R7,MINZER,2,2,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
450 MB = MDAT(M,1)
    MV = MDAT(M,2)
    CALL DATA (MV,1,1,7,V0)
    CALL REV MAT(M,MB,V0)
    RETURN

```

SW4.207
SW4.208
SW4.209
SW4.210
SW4.211
SW4.212
SW4.213
SW4.214
SW4.215

C

C CON ENTRY

```

50  IF (MODE.EQ.3) RETURN
    CALL INP8(M,0,0,2,2,2,1,0)
    RETURN

```

C

C VPAR ENTRY

```

55  IF (MODE.EQ.3) RETURN

```

```

      CALL INP6(M,0,0,KA,1,0)
      RETURN
C
C          NPOL ENTRY
C  Q      NPOL  N  J      L      CM      BRHO
56  IF (MODE.EQ.3) GO TO 561
      CALL INP6 (M,MAP,MINZER,3,1,0)
      IF (MODE.EQ.2) RETURN
      IF (XSW) RETURN
C  PRESET MATRIX = UNIT MATRIX
561 CALL RTRV7(MUNIT,T,RW)
      CALL STOR7(M,T,RW)
      RETURN
C
C  ORBC ENTRY
57  IF(MODE.EQ.3) RETURN
      CALL INP8 (M,0,0,4,2,2,1,0)
      IF(MODE.EQ.2) RETURN
      IF (XSW) RETURN
571 CALL ORBIT(M)
      RETURN
C
      END
C*****SW5*****
      SUBROUTINE SW5
      IMPLICIT REAL*8 (A-H,O-Z)

C  PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.
C  PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.
C
      DIMENSION INF(24)
      INCLUDE 'INSTR.CCC'
      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BWORK.CCC'
      INCLUDE 'STORE.CCC'
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'DIM.CCC'
      INCLUDE 'BMI.CCC'
      INCLUDE 'SWTCH.CCC'
      INCLUDE 'GRR.CCC'

      INTEGER BDAT,PVECS
      DIMENSION T(49),RW(3)
      DIMENSION PAR(100),KREM(100)
      INTEGER R7
      EQUIVALENCE (OPNAME,INF(1)),(PAR,KREM)
      DATA PVECS,MAT,R7/2,3,5/
      DATA ISCL/9/

C
*      IF OPNAME = NULL, IT IS A P OR C CARD.  NEEDS SPECIAL
*      HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION

```

```

C  CALC FUNCTION IF OP=123
    IF (OP.GT.123) GO TO 176
    IF (OP.EQ.123) GO TO 123
    NOP = OP - 99
    IF(OP.GT.111) GO TO 75
    GO TO (40,43,42,64,65,71,66,68,67,62,72,74), NOP
*  GO TO PBML,MAT,VEC,PVEC,MXV,CPLX,EQIL,EVEC,PRTV,PRV7,CVEC
C      ,LIST,SIN,COS,EXP,ALOG,SQ,SQRT,TAN,ASIN,ACOS,ATAN,ABS
C      ,CALC,FBLE,FBLI,BMLE,BMLI,MAT7
C
C  GO TO FBLE,FBLI,BMLE,BMLI,MAT7
176  NOP = OP -123
    GO TO (1244,1244,1245,1246,900), NOP
C  PBML ENTRY      PRINT BEAM LINE
40  IF (MODE.EQ.3) GO TO 401
    CALL INP6(M,0,0,1,-2,0)
    IF (MODE.EQ.2) RETURN
401 IF (XSW) RETURN
    CALL PBML(M)
    RETURN
C
C  VEC ENTRY — KB VECTORS EACH WITH KA COMPONENTS
42  IF (MODE.EQ.3) RETURN
    IF (KB.NE.0) GO TO 421
    KB = 1
    INFF(5,M) = KB
421 ROWS = KA
    COLS = KB
    MN = KA*KB
    CALL RESRV6(M,MAT,MINZER,
    .   MN,1,0)
    DO 420 J=1,KB
420  CALL LOAD4(M,KA,1,0)
    RETURN
C  MAT ENTRY—MATRIX OF KA ROWS, KB COLS (INPUT BY ROWS)
43  IF (MODE.EQ.3) GO TO 430
    MN = KA*KB
    CALL RESRV6(M,MAT,MN,
    .   MN,1,0)
431 DO 432 J=1,KA
432  CALL LOAD4(M,
    .   KB,1,0)
430  CALL TRNSPS
    RETURN
C  MAT7 ENTRY —7X7 MATRIX
900 IF (MODE.EQ.3) GO TO 910
    CALL RESRV6(M,R7,MINZER,49,1,0)
    DO 901 J=1,7
901  CALL LOAD4(M,7,1,0)
910  CALL DATA (M,1,1,49,PAR)
    DO 902 J =1,3
902  RW(J) = 0.
    CALL STOR7(M,PAR,RW)
    RETURN

```

C

C PRINT PARTICLE VECTORS

```

62  IF (MODE.EQ.3) GO TO 620
    CALL INP6(M,0,0,
      KA,2,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
620  CALL PRNTV7(M,KA)
    RETURN

```

C PVEC ENTRY. READ IN KA VECTORS (COL) OF 6 ROWS. SET 7TH TO 1.0

```

64  IF(MODE.EQ.3) RETURN
    IF (KA.EQ.0) KA = 1
    V7 = 0.0
    DECODE (4,641,ICARD(5))KH
641  FORMAT (A3,1X)
    IF (KB.EQ.1.OR.KH.EQ.1H ) V7=1.D0

```

C PVECS KIND RESERVES 7 X KA

```

    KC = 7*KA
    COLS = KA
    CALL RESRV6(M,PVECS,MINZER,KC,1,0)
    DO 640 I=1,KA
    CALL LOAD4(M,
      6,1,0)
    NF = NF + 1
    NPL = 7*I
    CALL REPFLT(M,NPL,V7)
640  CONTINUE
    RETURN

```

C MXV ENTRY. MATRIX TIMES VECTOR

```

65  IF(MODE.EQ.3) GO TO 650
    ROWS = 7
    COLS = 1
    CALL INP6(M,PVECS,7,2,2,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN

```

C RETRIEVE ELEMENTNUMBER OF INPUT MATRIX AND VECTOR.FIND LQ OF VECTORS

```

650  M1 = MDAT(M,1)
    NM = BDAT(M,2)
    M2 = ELNUM(NM)
    LOC = 1
    IF(.NOT.VCSW) GO TO 651
    DECODE (4,652,NM) NUM
652  FORMAT (1X,I1,2X)
    LOC = (NUM-1)*7+1
651  CALL DATA (M2,5,LOC,7,PAR(8))
    CALL RTRV7(M1,T,RW)
    CALL MXV7(T,PAR(8),PAR(1))
    CALL STDAT(M,5,1,7,PAR)
    RETURN

```

C EQIL ENTRY

```

66  IF (MODE.EQ.3) GO TO 660
    KA = 3
    CALL INP6(M,PVECS,MINZER,
      2,2,0)

```

C SET KA FOR NUMBER OF COLUMN VECTORS

IF (MODE.EQ.2) RETURN

IF (XSW) RETURN

660 CALL EQIL(M)

RETURN

C PRINT VECTORS ANY SIZE

67 IF(MODE.EQ.3) GO TO 670

CALL INP6(M,0,0,

. KA,2,0)

IF (MODE.EQ.2) RETURN

IF (XSW) RETURN

670 CALL PRINTV(M,KA)

RETURN

C EVEC ENTRY

68 IF (MODE.EQ.3) GO TO 680

CALL INP6(M,0,112,

. 2,2,0)

IF (MODE.EQ.2) RETURN

IF (XSW) RETURN

680 CALL EVEC(M)

RETURN

C CPLX ENTRY —A COMPLEX NUMBER

71 IF (MODE.EQ.3) RETURN

TYPE = 4HCPLX

KN = ISCL

CALL INP6(M,KN,0,

. 2,1,0)

RETURN

C CVEC ENTRY —COMPLEX PARTICLE VECTOR,KA, IN NUMBER

C IF KB=BLANK OR 1, ROW 7 IS 1. IF KB = 0, ROW7 IS 0.

72 IF (MODE.EQ.3) RETURN

V7R = 0.

V7I=0.

IF (KB.EQ.1.OR.KB.EQ.1H) V7R = 1.

IF(KA.EQ.0) KA=1

KC = 14*KA

ROWS = 7

COLS = KA

TYPE = 4HCPLX

CALL RESRV6(M,PVECS,0,

. KC,1,0)

DO 721 I=1,KA

CALL LOAD4(M,

. 12,1,0)

NF = NF + 2

NPL = 14*(I-1) + 13

CALL REPFLT(M,NPL,V7R)

NPL = NPL + 1

CALL REPFLT (M,NPL,V7I)

721 CONTINUE

RETURN

C LIST ENTRY

C

C A LIST KA A1 A2 A3 A4 A5

C

C THE LIST INSTRUCTION SPECIFIES A SET OF ELEMENTS THAT ARE TO BE
C USED SUCCESSIVELY. THE NAME APPEARS IN A BML INSTRUCTION. EACH
C TIME THE SAME LIST NAME IS ENCOUNTERED, THE NEXT AI IS USED.

C

C EXAMPLE

C

C	QQ	LIST	Q1	Q2	Q3	Q4	Q5
C	L	DRF	3.				
C	.S	BML	5(Q	L		)	

C

C .S IS EQUIVALENT TO

C	.S	BML	Q1	L	Q2	L	Q3	L	Q4	L	Q5	L
---	----	-----	----	---	----	---	----	---	----	---	----	---

C

```

74  IF (MODE.EQ.3) GO TO 741
    CALL INP6(M,0,0,1,-2,0)
741 IF (KB.NE.0) RETURN
    KB=1
    INFF(5,M)=1
    RETURN

```

C SCALAR FUNCTIONS

```

75  IF (MODE.EQ.3) GO TO 750
    KN = ISCL
    CALL INP6(M,KN,1,1,1,0)
    IF (MODE.EQ.2.OR.XSW) RETURN
750 XX = FDAT(M,1)
    NNOP = OP - 111
    GO TO (76,77,78,79,80,81,82,83,84,85,86), NNOP
76  YY = DSIN(XX)
    GO TO 99
77  YY = DCOS(XX)
    GO TO 99
78  YY = DEXP(XX)
    GO TO 99
79  YY = DLOG(XX)
    GO TO 99
80  YY = XX*XX
    GO TO 99
81  YY = DSQRT(XX)
    GO TO 99
82  YY = DTAN(XX)
    GO TO 99
83  YY = DASIN(XX)
    GO TO 99
84  YY = DACOS(XX)
    GO TO 99
85  YY = DATAN(XX)
    GO TO 99
86  YY = DABS(XX)
99  CALL STDAT(M,5,1,1,YY)
    RETURN

```

C CALC ENTRY

```

123 IF (MODE.EQ.3) GO TO 1231

```

```

      CALL INP6(M,0,5,1,-2,0)
C   SET LFL AND NFL EQUAL TO LQ AND NQ
      INFF(M,14) = INFF(M,10)
      INFF(M,15) = INFF(M,11)
      IF (MODE.EQ.2.OR.XSW) RETURN
1231 CALL CALC(M)
      RETURN
C
C   ***** FBLE AND FBFI ENTRIES *****
C   FBLE —EXPLICIT BEAM LINE FUNCTION. WHEN REFERRED TO BY
C   BMLE, A NEW BML STATEMENT IS CREATED.
C   NM FBLE K T A1 A2 A3 ... AK
C   NM = NAME OF FUNCTION
C   T = NAME OF BEAM LINE TO BE CHANGED.
C   K = NUMBER OF REPLACEMENTS
C   A1,A2,...AK = ELEMENTS WHICH WILL BE REPLACED.
C   FBLE WILL BE FOLLOWED AT SOME TIME BY ONE OR MORE BMLE STATE-
C   MENTS CONTAINING REPLACEMENT ELEMENTS.
CC
C
C   FBFI —IMPLICIT BEAM LINE FUNCTION. REPLACEMENTS CALLED FOR
C   BY AN BMLI WILL NOT BE MADE UNTIL TIME FOR CREATING MILIST.
C   NM FBFI K T A1 A2 A3 ... AK
C   INPUT IS SAME AS FOR FBLE.
CC
C
1244 IF (MODE.EQ.3) RETURN
      CALL INP8(M,0,0,1,2,KA,2,0)
      RETURN
C
C
C   BMLE — REPLACE BEAM LINE ELEMENTS NAMED IN FUNCTION FBLE
C   WITH ONES NAMED HERE AND CREATE A NEW BEAM LINE
C   RE BMLE K FN BNAM B1 B2 B3 ... BK
C   FN = NAME OF BEAM LINE FUNCTION FBLE
C   BNAM = NAME OF NEW BEAM LINE. IF BLANK, IT WILL ASSUME NAME OF
C   THE OLD ONE NAMED IN FUNCTION.
C   K = NUMBER OF REPLACEMENTS (MUST BE THE SAME AS K IN FUNCTION FBLE)
C   B1,B2,B3...BK = ELEMENTS THAT REPLACE A1,A2,...NAMED IN FUNCTION.
C
1245 IF (MODE.EQ.3) RETURN
      KT = KA + 2
      CALL INP6(M,0,0,KT,2,0)
      MFN = MDAT(M,1)
      NAMOP = INFF(1,MFN)
      IF (NAMOP.NE.'FBLE') GO TO 245
      CALL BMLE(M)
      RETURN
245 WRITE (3,246) NAME,NAMOP
246 FORMAT (5X,'*** ERROR *** BMLE STATEMENT ',A5,2X,'REFERS TO ',
1 A5,' INSTEAD OF AN FBLE.')
      CALL EXIT
C   BMLI ENTRY
1246 IF (MODE.EQ.3) RETURN

```


C BMLI — REPLACE ELEMENTS NAMED IN FB LI INSTRUCTION WITH THOSE
 C INPUT HERE. DONE AT TIME MILIST IS CREATED. CHANGES
 C MAY THREAD THROUGH EMBEDDED BEAM LINES, BUT NO OTHER
 C BMLI'S MAY BE INCLUDED.

C BL BMLI K FN B1 B2 B3 ... BK

C FN = NAME OF BEAM LINE FUNCTION FB LI

C K = NUMBER OF REPLACEMENTS (SAME AS FOR FB LI)

C B1,B2,...BK ARE CONSTITUTENTS THAT REPLACE A1,A3,... NAMED IN

C FN (FB LI INSTRUCTION).

C

C .P BML A B C C

C .Q BML E F .P

C FQ FB LI 2 .Q A C

C .QQ BMLI 2 FQ A1 C1

C CYC .QQ

C

KT = KA + 1

CALL RESRV6(M,0,0,KT,2,0)

CALL LOAD6(M,1,2,KA,2,0)

MFN = MDAT(M,1)

NAMOP = INFF(1,MFN)

IF (NAMOP.EQ.'FB LI') THEN

KF = INFF(4,MFN)

KR = INFF(4,M)

IF (KF.EQ.KR) RETURN

WRITE (3,248)

248 FORMAT (5X,'*** ERROR *** NUMBER OF REPLACEMENT ITEMS DIFFERS '

1 'IN FB LI AND BMLI STATEMENTS.')

CALL EXIT

ELSE

WRITE (3,247) NAME,NAMOP

247 FORMAT (5X,'*** ERROR *** BMLI STATEMENT ',A5,2X,'REFERS TO ',

1 A5,' INSTEAD OF AN FB LI.')

CALL EXIT

END IF

END

C ***** SW6 *****

SUBROUTINE SW6

IMPLICIT REAL*8 (A-H,O-Z)

C PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.

C PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.

C

DIMENSION INF(24)

INCLUDE 'INSTR.CCC'

INCLUDE 'STORE.CCC'

INCLUDE 'DIM.CCC'

INCLUDE 'BMI.CCC'

INCLUDE 'SWTCH.CCC'

INCLUDE 'CONTRL.CCC'

INCLUDE 'GRR.CCC'

EQUIVALENCE (OPNAME,INF(1))

DATA MXP MY/1/

C
 C IF OPNAME = NULL, IT IS A P OR C CARD. NEEDS SPECIAL
 C HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION

NOP = OP - 150
 GO TO (76,77,78,79,83,84,85,86,87,88), NOP

C GO TO BEAM, DKE, EDRF, SOL, CYEM, BVAL, CYAE,
 C IBS,UPIO,PARA

C BEAM ENTRY. DEFINE A BEAM PARTICLE

76 CONTINUE
 IF (MODE.EQ.3) GO TO 761
 CALL RESRV8(M,0,10,8,1,1,3,0)

C LOAD 4 COMPONENTS OF VECTOR FROM CARD 1
 CALL LOAD4(M,4,1,0)

C LOAD SECOND CARD
 CALL LOAD4(M,4,1,0)
 IF (MODE.EQ.2) RETURN
 IF (XSW) RETURN

761 CALL BEAM(M)
 RETURN

C DKE ENTRY. ENERGY GAIN

77 CONTINUE
 IF (MODE.EQ.3) RETURN
 CALL INP6(M,0,0,1,1,0)
 RETURN

C EDRF ENTRY.

78 CONTINUE
 IF (MODE.EQ.3) RETURN
 CALL INP6(M,0,0,1,1,0)
 RETURN

C SOL ENTRY

79 CONTINUE
 IF (MODE.EQ.3) RETURN
 CALL INP6(M,0,0,3,1,0)
 RETURN

C

C CYEM ENTRY

83 IF (MODE.EQ.3) GO TO 831
 CALL INP8(M,0,4,1,2,5,1,0)
 IF (MODE.EQ.2) RETURN
 IF (XSW) RETURN
 831 CALL CYEM(M)
 RETURN

C BVAL ENTRY

84 IF (MODE.EQ.3) GO TO 841
 CALL INP6(M,0,4,6,1,0)
 IF (MODE.EQ.2) RETURN
 IF (XSW) RETURN
 841 CALL BVAL(M)
 RETURN

C CYAE ENTRY

```

85  IF (MODE.EQ.3) GO TO 851
    CALL INP8(M,0,0,2,2,3,1,0)
    IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
851 CALL CYAE(M)
    RETURN

```

C IBS ENTRY

```

86  IF (MODE.EQ.3) GO TO 861
    CALL INP(M,MXPMY,MINZER,4,2,4,4,6,1,0)
    IF (MODE.EQ.2.OR.XSW) RETURN
861 CALL IBS(M)
    RETURN

```

C UPIO ENTRY. WRITES NEW SYNCH DATA FILE WITH UPDATED VALUES.

```

87  KA2 = KA*2
    CALL INP6(M,0,0,KA2,2,0)
    CALL UPIO(M)
    RETURN

```

C PARA ENTRY

```

88  IF (MODE.EQ.3) GO TO 881
    KN = IBLNK

```

C KIND AND TYPE ARE SCALAR AND SINGLE

```

    KN = ISCL
    CALL INP8(M,KN,0,
      1,1,1,3,0)
881  IF (XSW) RETURN
    IEXP = IDAT(M,1)
    FEXP = 1
    IF (IEXP.EQ.0) GO TO 882
    FEXP = 10.0**IEXP
882  F = FDAT(M,1)
    FF = F * FEXP
    CALL STDAT(M,1,1,1,FF)
    RETURN
    END

```

C***** SW7 *****

SUBROUTINE SW7

IMPLICIT REAL*8 (A-H,O-Z)

C PROCESSES DATA FOR CURRENT INSTR. IF IN MODE 1 OR 2.

C PROCEEDS WITH CALC. BY CALLING APPROPRIATE ROUTINES IF MODE=1 OR 3.

C

```

    DIMENSION INF(24)
    INCLUDE 'INSTR.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'STORE.CCC'
    INCLUDE 'CONTRL.CCC'
    INCLUDE 'DIM.CCC'
    INCLUDE 'BMI1.CCC'
    INCLUDE 'SWTCH.CCC'
    INCLUDE 'GRR.CCC'

```

```

    DIMENSION PAR(100),KREM(100)
    INTEGER R7,ROT,PRD

```

```

INTEGER BDAT
EQUIVALENCE(OPNAME, INF(1)),(PAR,KREM)
DIMENSION V0(7)
DATA R7,ROT,MAP,PRD/5,6,7,8/
DATA RADEG/.0174532925/
DATA V0/7*0./
DATA IBLNK,MXPMY,PVECS,R7,ROT,PRD/'      ',1,2,5,6,8/

C
*      IF OPNAME = NULL, IT IS A P OR C CARD.  NEEDS SPECIAL
*      HANDLING TO CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION
C
NOP = OP - 160
GO TO (36,37,38,39,40,41) NOP
C
GO TO TRKB,SOLV,SMIN,TRKE,TRKM,IBET
C
C TRKB ENTRY
36  IF (MODE.EQ.3) GO TO 360
    CALL RESRVA(M,MXPMY,MINZER,
      . 4,2,6,3,2,1,0,0
    .)
    CALL LOAD8(M,
      . 4,2,4,3,2,1,0)
    CALL STDAT(M,3,6,1,0)
C  6TH INTEGER IS A FLAG TO TELL TRKB IF IT HAS BEEN CALLED YET.
    IP1 = IDAT(M,1)
    IP2 = IDAT(M,2)
    NAMVEC = BDAT(M,3)
    NAMOP = INFF(1,M)
    NCOL = 20
    IF(NAMVEC.EQ.IBLNK) NCOL = 14

    CALL STDAT(M,3,5,1,NCOL)
    MBML = MDAT(M,1)
    NDAT = INFF(17,MBML)
    CALL MIFILL(MBML,1,NDAT,NEL,MI1)
    IF (KA.NE.0.OR.KB.NE.0) GO TO 361
    NQ3 = NCOL*(NEL+1)
    KB = NEL
    GO TO 362
361  IF (KA.LE.-1) GO TO 363
    NQ3 = NCOL*(KB-KA+1)
362  LQ3 = LFILE + 1
    LFILE = LQ3 + NQ3
    NTOT = NTOT + NQ3
    INFF(5,M) = NEL
    INFF(18,M) = NTOT
    INFF(24,M) = LQ3
363  IF (IP1.NE.0.OR.IP2.NE.0) GO TO 364
    CALL REPINT(M,1,0)
    CALL REPINT(M,2,NEL)
364  IF (MODE.EQ.2) RETURN
    IF (XSW) RETURN
360  CALL TRKB(M)
    RETURN
C

```

SW4.19

SW4.20

SW4.21

SW4.22

SW4.23

SW4.24

SW4.25

SW4.26

SW4.71

SW4.72

SP9SW4.2

SW4.76

SW4.77

SP9SW4.3

SW4.80

SW4.81

SW4.86

C	TRKE ENTRY	SW4.71
39	IF (MODE.EQ.3) GO TO 390	SW4.72
	CALL RESRVA(M,MXPMY,MINZER,	
	. 4,2,6,3,2,1,0,0	SP9SW4.2
	.)	SW4.76
	CALL LOAD8(M,	SW4.77
	. 4,2,4,3,2,1,0)	SP9SW4.3
	CALL STDAT(M,3,6,1,0)	SW4.80
C	6TH INTEGER IS A FLAG TO TELL TRKE IF IT HAS BEEN CALLED YET.	SW4.81
	IP1 = IDAT(M,1)	
	IP2 = IDAT(M,2)	
	NAMVEC = BDAT(M,3)	
	NAMOP = INFF(1,M)	
	NCOL = 20	
	IF(NAMVEC.EQ.IBLNK) NCOL = 14	
	CALL STDAT(M,3,5,1,NCOL)	
	MBML = MDAT(M,1)	
	NDAT = INFF(17,MBML)	
	CALL MIFILL(MBML,1,NDAT,NEL,M11)	
	IF (KA.NE.0.OR.KB.NE.0) GO TO 391	
	NQ3 = NCOL*(NEL+1)	
	KB = NEL	
	GO TO 392	
391	IF (KA.LE.-1) GO TO 393	
	NQ3 = NCOL*(KB-KA+1)	
392	LQ3 = LFILE + 1	
	LFILE = LQ3 + NQ3	
	NTOT = NTOT + NQ3	
	INFF(5,M) = NEL	
	INFF(18,M) = NTOT	
	INFF(24,M) = LQ3	
393	IF (IP1.NE.0.OR.IP2.NE.0) GO TO 394	
	CALL REPINT(M,1,0)	
	CALL REPINT(M,2,NEL)	
394	IF (MODE.EQ.2) RETURN	
	IF (XSW) RETURN	SW4.86
390	CALL TRKE(M)	
	RETURN	
C		
C	TRKM ENTRY	SW4.71
40	IF (MODE.EQ.3) GO TO 400	SW4.72
	CALL RESRVA(M,MXPMY,MINZER,	
	. 4,2,6,3,2,1,0,0	SP9SW4.2
	.)	SW4.76
	CALL LOAD8(M,	SW4.77
	. 4,2,4,3,2,1,0)	SP9SW4.3
	CALL STDAT(M,3,6,1,0)	SW4.80
C	6TH INTEGER IS A FLAG TO TELL TRKM IF IT HAS BEEN CALLED YET.	SW4.81
	IP1 = IDAT(M,1)	
	IP2 = IDAT(M,2)	
	NAMVEC = BDAT(M,3)	
	NAMOP = INFF(1,M)	
	NCOL = 20	
	IF(NAMVEC.EQ.IBLNK) NCOL = 14	
	CALL STDAT(M,3,5,1,NCOL)	

```

MBML = MDAT(M,1)
NDAT = INFF(17,MBML)
CALL MIFILL(MBML,1,NDAT,NEL,MI1)
IF (KA.NE.0.OR.KB.NE.0) GO TO 401
NQ3 = NCOL*(NEL+1)
KB = NEL
GO TO 402
401 IF (KA.LE.-1) GO TO 403
NQ3 = NCOL*(KB-KA+1)
402 LQ3 = LFILE + 1
LFILE = LQ3 + NQ3
NTOT = NTOT + NQ3
INFF(5,M) = NEL
INFF(18,M) = NTOT
INFF(24,M) = LQ3
403 IF (IP1.NE.0.OR.IP2.NE.0) GO TO 404
CALL REPINT(M,1,0)
CALL REPINT(M,2,NEL)
404 IF (MODE.EQ.2) RETURN
IF (XSW) RETURN
400 CALL TRKM(M)
RETURN

```

SW4.86

C IBET ENTRY

```

41 IF (MODE.EQ.3) RETURN
CALL INP6(M,0,0,12,1,0)
RETURN

```

C SOLV ENTRY

```

37 IF (MODE.EQ.3) GO TO 370
NF1 = 1 + 2*KA + 3*KB
C NB1 = 4 + 2*KA + 6*KB
C NI1 = 5 + 6*KA
NB1 = 4 + 8*KA + 6*KB
NI1 = 5
CALL RESRVA(M,0,0,
NF1,1,NB1,2,NI1,3,0)
CALL LOAD8(M,4,2,5,3,1,1,0)
DO 371 I=1,KA
CALL LOAD6(M,
.8,2,2,1,0)
371 CONTINUE
372 DO 373 J=1,KB
CALL LOAD6(M,
.6,2,3,1,0)
373 CONTINUE
IF (MODE.EQ.2) RETURN
IF (XSW) RETURN
370 CALL SOLV(M)
RETURN

```

SW4.92

SW4.95

SW4.96

SW4.99

SW4.100

SW4.103

SW4.104

SW4.105

SW4.106

SW4.108

SW4.109

SW4.110

SW4.112

SW4.113

C SMIN ENTRY

```

38 IF (MODE.EQ.3) GO TO 380
NF1 = 5*KA
NB1 = 3*KA

```

```

CALL RESRV8(M,0,0,NF1,1,NB1,2,0)
K = 1
DO 381 I=1,KA
CALL LOAD6(M,1,4,5,1,0)
KREM(1) = ICARD(6)
KREM(2) = ICARD(7)
DECODE(2,382,ICARD(8))KREM(3)
382 FORMAT (A2)
CALL STDAT(M,2,K,3,KREM)
K = K + 3
381 CONTINUE
IF (XSW) RETURN
380 CALL SMIN(M)
RETURN
END
C ***** SWITCH *****
SUBROUTINE SWITCH
IMPLICIT REAL*8 (A-H,O-Z)

C CONTROLS CALLS TO DIFFERENT SWITCHES OR OVERLAYS, ACCORDING TO
C OP OF SYNCH INSTRUCTION.
INCLUDE 'INSTR.CCC'
INCLUDE 'CONTRL.CCC'
INCLUDE 'GRR.CCC'

C IF OPNAME=NULL, IT IS A P OR C CARD. IT NEEDS SPECIAL HANDLING TO
C CONSTRUCT A SYNCH PAGE OR REM INSTRUCTION.
IF (OPNAME.NE. 4HNULL) GO TO 5
CALL SPLOAD
RETURN
5 CONTINUE
XSW = .FALSE.

C IF OP IS NEGATIVE, EXECUTION IS TO BE DELETED. TURN ON XSW.
IF (OP.LT.0) XSW = .TRUE.
IF ((OP.LE.0).AND.(MODE.EQ.3)) RETURN

C TURN OP TO POSITIVE IF LOADING IS REQUIRED
IF (XSW) OP = -OP

C IF OP WAS ILLEGAL, IT WAS SET TO 0 BY OPNUM. ERROR SWITCH IS ON
IF (OP.EQ.0) RETURN

C IF OP IS FIN, INCR OR REPL, IGNORE ERROR. GO TO NEXT CASE.
IF (OP.EQ.2.OR.OP.EQ.7.OR.OP.EQ.8) IERR = .FALSE.
IF (IERR) RETURN
IERR = .FALSE.
IF (OP.GT.50) GO TO 10
CALL SW1
GO TO 20
10 IF (OP.GT.64) GO TO 11
CALL SW3
GO TO 20
11 IF (OP.GE.100) GO TO 12
CALL SW4
GO TO 20
12 IF (OP.GT.150) GO TO 14
CALL SW5
GO TO 20

```

```

14  IF (OP.GT.100) GO TO 15
    CALL SW6
    GO TO 20
15  IF (OP.GT.200) GO TO 16
    CALL SW7
20  RETURN
C      IF OP.GT.200 IT IS AN ERROR
16  WRITE(3,50) OP,OPNAME
50  FORMAT (5X,3H***,15,' IS AN INVALID OP NUMBER.    SYNCH INSTRUCTIO
1N NAMED 'A4)
    ERROR = .TRUE.
    RETURN
    END

```

C ***** SXTTP *****

```

    SUBROUTINE SXTTP(M,V,PAR)
C  SX  SXTTP    KB  L    D2B        BRHO        EXACT
C  _____
C    SX IS A SEXTUPOLE THAT ACTS AT A POINT.  IT REPRESENTS THE
C  EFFECT OF A SEXTUPOLE OF LENGTH L, WITH D2B = D2B/DX2.
C  IF L=0 ON INPUT, L IS SET =1.
C    EXACT IS AN OPTION TO CALL FOR AN EXACT CALCULATION, USING
C  ELLIPTIC FUNCTIONS ACCORDING TO A METHOD OF B. AUTIN. IT IS
C  EXERCISED BY SETTING EXACT TO A NON-ZERO VALUE, BUT IS ONLY
C  OBEYED IF THE RAY IS IN THE HORIZONTAL PLANE AND L IS NOT ZERO.
C  _____
C
C    IF KB NOT 0, SETS R(2,6) = R(4,6) = 0.
C
C  IF KA.NE.0 SEXTUPOLE IS REPLACED BY GENERALIZED MULTIPOLE
C  OF ORDER KA, SKEW IF KB=1 (SEE NPOL WRITEUP)

```

```

    IMPLICIT REAL*8 (A-H,O-Z)
    INCLUDE 'DIM.CCC'
    INCLUDE 'BINFF.CCC'
    COMMON/MP/ML(3)

```

```

    DIMENSION V(1),PAR(1),W(4),P(4),R(7,7),RW(3),MAG(6)
    EQUIVALENCE (X,W(1)),(PX,W(2)),(Y,W(3)),(PY,W(4)),
1      (L,P(1)),(D2B,P(2)),(BRHO,P(3))
2      ,(LSX,MAG(1)),(K,MAG(2)),(ONE,MAG(3)),(CURV,MAG(4)),
3      (EPS0,MAG(5)),(EPS1,MAG(6))
    REAL*8 L,LSX,LH,K,KP,MAG
    DATA ONE/1.D0/,S3/1.732050808D0/,S32/0.433012702D0/
    DATA SS3/1.44224957D0/,S13/.33333333D0/,S23/3.464101616D0/
    INCLUDE 'CONST.CCC'

```

```

C
    IF (INFF(4,M).EQ.0) GO TO 15
    CALL NPOL(M,V,PAR)
    RETURN
15  CONTINUE
    DO 1 I=1,4
    W(I)=V(I)
1  P(I)=PAR(I)
    DPP = V(6)

```



```

FAC = 1.D0 + DPP
BR = FAC * BRHO
F = D2B / BR
KP=F
C IF KB.NE.0, TURN OFF COLUMN 6
KB = INFF(5,M)
IF(X.EQ.0..AND.PX.EQ.0..OR.KP.EQ.0.) GO TO 11
IF(P(4).NE.0..AND.L.NE.0..AND.Y.EQ.0..AND.PY.EQ.0.) GO TO 5
11 IF (L.EQ.0.) GO TO 2
LH = L/2.D0
X = X + PX * LH
Y = Y + PY * LH
F = L * F

2 CONTINUE
DPX=-(F/2.D0)*(X*X-Y*Y)
DPY=F*X*Y
V(2) = V(2) + DPX
V(4) = V(4) + DPY
IF (L.EQ.0) GO TO 3
V(1) = X + V(2)*LH
V(3) = Y + V(4)*LH

3 IF (M.EQ.0) RETURN
CALL RTRV7(MUNIT,R,RW)
T21 = - F * X
T23 = F * Y
R(2,1) = T21
R(2,3) = T23
R(4,1) = T23
R(4,3) = - T21
R(2,6)=-DPX
R(2,7)=-DPX
R(4,6)=-DPY
R(4,7)=-DPY
R(5,1)=DPX
R(5,3)=DPY
IF (KB.EQ.0) GO TO 7
R(2,6) = 0.
R(4,6) = 0.
7 RW(2) = -DPX

IF (L.EQ.0.) GO TO 4
ML(1)=MEND-3
ML(2)=M
ML(3)=MEND-3
CALL DRIFT(MEND-3,LH)
CALL STOR7(M,R,RW)
CALL MM(M,3,ML)
RETURN

4 CONTINUE
CALL STOR7(M,R,RW)
RETURN

```

```

C  EXACT CALCULATION FOR THICK SEXTUPOLE, H-PLANE ONLY. — B.AUTIN
5  CONTINUE
C
C  CONDITIONS INITIALES C
C
  SK=DSIGN(1.D0,KP)
  AK=(DABS(KP))*S13
  U=KP*PX
  SU=-DSIGN(1.D0,U)
  C=X*X*X*KP/3.D0+PX*PX
  IF(C.EQ.0.) GO TO 99
  AC=DABS(C)**S13
  SC=DSIGN(1.D0,C)
  AXM=AC*SS3/AK
  XM=SC*SK*AXM
  TM=DSQRT(S3*AC)*AK/SS3
C
C  MODULE K2
C
  XK2=.5D0+SC*S32
  XK=DSQRT(XK2)
C
C  LIGNES TRIGONOMETRIQUES DE PHI0
C
  T=DABS(1.D0-X/XM)/S3
  D=1.D0+T
  CT=(1.D0-T)/D
  ST=SU*2.D0*DSQRT(T)/D
  ST2=ST*ST
  DT=DSQRT(1.D0-XK2*ST2)
C
C  FONCTIONS DE JACOBI
C
  CALL QFUNCT(XK,XKPRIM,Q)
  XK=DSQRT(XK)
  XKP=DSQRT(XKPRIM)
  Z=TM*L
  TH3=THETA3(0.,Q)
  U=Z/(TH3*TH3)
  TH4=THETA4(U,Q)
  XT=XK*TH4
  SN=THETA1(U,Q)/XT
  SN2=SN*SN
  CN=XKP*THETA2(U,Q)/XT
  DN=XKP*THETA3(U,Q)/TH4
C
C  X1
C
  C1=(CT*CN-ST*SN*DT*DN)/(1.D0-XK2*ST2*SN2)
  XX=SC*S3*(1.D0-C1)/(1.D0+C1)
  X1M=1.D0-XX
  X1=X1M*XM
C
C  XPRIME1
C

```

```

S1=ST*CN*DN+CT*SN*DT
U=KP*S1
SU=-DSIGN(1.D0,U)
CX=DABS(C*XX)
PX1=SU*DSQRT(CX*(1.D0+X1M+X1M*X1M))
GO TO 100

C
C   CAS PARTICULIER C=0
C
99  XM=-SK/AK
    TM=AK/S23
    T=SU*DSQRT(X/XM)
    T1=T/(1.D0-TM*T*L)
    T12=T1*T1
    X1=XM*T12
    PX1=-SK*T12*T1/S3
100 V(1)=X1
    V(2)=PX1

    IF(M.EQ.0) RETURN
C   MAG(I) = LSX,K,1.0,CURV,EPS0,EPS1 — INPUT TO S/R MAGNET
    LSX=L
    THETA=PX-PX1
    XAV=(X+X1)/2.D0 + THETA*L/12.D0
    K=KP*XAV
    CURV=THETA/L
    EPS0=PX/RADEG
    EPS1=-PX1/RADEG
    CALL MAGNET(M,MAG)

    RETURN
    END
C   ***** SYSTM C *****
    SUBROUTINE SYSTM C
    RETURN
    END
C   ***** TABULT *****
    SUBROUTINE TABULT(M,TABSW)
    IMPLICIT REAL*8 (A-H,O-Z)
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'BWORK.CCC'
    DIMENSION ISTORE(1)
    EQUIVALENCE (STORE(1),ISTORE(1))
    INCLUDE 'SWTCH.CCC'

    DIMENSION INDX(20),NAM(10),JND(20),F(10),IVAR(2)
    LOGICAL TABSW
    INTEGER ELNUM,RIGHT,BDAT

    IVAR(1)=0
    KK=KA*2
    DO 50 I=1,KK
        INDX(I)=0
50    JND(I)=0

```

```

      KA = INFF(4,M)
      KB = INFF(5,M)
      KS = KA*60
C   IF KB.GT.0, SET IPR=KB, WHICH GOES TO INNF(6)
C   OF CYX INSTRUCTION TO SUPPRESS PRINT.
      IPR = 0
      IF (KB.GT.0) IPR = KB
      IF (TABSW) GO TO 120
C   FETCH BCD LIST
C   SET UP TABLE OF INDICES
      CALL DATA(M,2,1,KA,NAM)
      DO 100 I=1,KA
      NM = NAM(I)
      LM = ELNUM(NM)
      JND(I*2) = 1
      INDX(I*2) = LM
      IF (LM.NE.0) INFF(6,LM) = IPR
      IF (.NOT.VCSW) GO TO 100
C   WHEN VCSW IS ON, ELEMENT IS ONE OF A VECTOR
C   SHIFT OFF LETTER NAME. SHIFT RIGHT FOR INTEGER NUMBER
      LNUM = SHFTL(NM,8)
      LNUM = SHFTR(LNUM,24)
      JND(I*2)=LNUM
100  CONTINUE
      CALL STDAT(M,7,1,KA,INDX)
      CALL STDAT(M,8,1,KA,JND)
      RETURN
120  CONTINUE
      LQ = INFF(10,M)
C   GET INDECEES
      CALL DATA(M,7,1,KA,INDX)
      CALL DATA(M,8,1,KA,JND)
C   GET CURSOR FROM LQ
      FVAR=STORE(LQ)
      IC=FVAR
      IUP = IC + KA
      IF (IUP.GT.KS) GO TO 130
      DO 125 I=1,KA
C   INDEX FOR VECTOR WILL NOT YET BE SET IF ORIGINALLY DEFINED IN SUB
      IF (INDX(I*2).NE.0) GO TO 124
      NM = BDAT(M,I)
      LM = ELNUM(NM)
      CALL REPLQ(M,2,I,INDX(I))
      INFF(6,LM) = IPR
      IF (.NOT.VCSW) GO TO 124
      LNUM = SHFTL(NM,8)
      DECODE(4,1,LNUM)JND(I*2)
1   FORMAT (I1,3X)
      CALL REPLQ(M,3,I,JND(I))
124  IN = INDX(I*2)
      JN = JND(I*2)
C   FETCH VALUE
      CALL FLQ(IN,2,JN,F(I))
125  CONTINUE
C   STORE

```

```

      CALL STDAT(M,5,IC,KA,F)
C  RESET CURSOR
      FVAR=IUP
      STORE(LQ)=FVAR
      RETURN
C  STORAGE FILLED.  PRINT TABLE AND CONTINUE
130  CALL PRNTAB(M)
      GO TO 120
      END

C ***** TINV7 *****
      SUBROUTINE TINV7(TINV,T)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TINV(7,7),T(7,7), STR(6,6),U(6,6),SC(6,6),VK(6)
C  INVERSE OF 7X7 SYMPLECTIC MATRIX
C  6X6 MATRIX  INVERSE OF S = SC = -U*STR*U
C              VK = (T(I,7),I=1,6)
C
C  7X7 MATRIX  INVERSE OF T = (SC-SC*VK)
C              (0   1 )
C
C  STR IS TRANSPOSE OF 6X6 OF  T
      DO 10  I = 1,6
      DO 10  J = 1,6
      STR(I,J) = T(J,I)
10  CONTINUE
C  SET UP  U ARRAY
      DO 15  J = 1,6
      DO 15  I = 1,6
      U(I,J) = 0.0
15  CONTINUE
      DO 20  I = 1,5,2
      U(I,I+1) = -1.
      U(I+1,I) = +1.
20  CONTINUE
      CALL MXM6 (STR,U,SC)
      CALL MXM6 (U,SC,SC)
      DO 25  J = 1,6
      DO 25  I = 1,6
      TINV(I,J) = - SC(I,J)
25  CONTINUE
C  SET UP VECOTR VK
      DO 30  I = 1,6
      VK(I) = T(I,7)
30  CONTINUE
      CALL MXV6(SC,VK,VK)
      DO 35  I = 1,6
      TINV(I,7) = VK(I)
      TINV(7,I) = 0.0
35  CONTINUE
      TINV(7,7) = 1.0
      RETURN
      END

C ***** THETA1 *****
      FUNCTION THETA1(U,Q)
      IMPLICIT REAL*8 (A-H,O-Z)

```

```

A=DSIN(U)
GO TO 1
ENTRY THETA2(U,Q)                                PSL
A=DCOS(U)
1 QN=2.*Q**.25
  THETA1=QN*A
  IF(THETA1.EQ.0.) RETURN
  C=(A+A)**2-2.
  B=A
  QM=1.
  GO TO 3
  ENTRY THETA3(U,Q)                                PSL
  A=DCOS(U+U)
  GO TO 2
  ENTRY THETA4(U,Q)                                PSL
  A=-DCOS(U+U)
2 C=A+A
  B=C*A-1.
  QM=Q**3
  QN=QM*(Q+Q)
  THETA1=QN*B+Q*C+1.
  IF(Q.EQ.0.) RETURN
3 QQ=DMIN1(Q**2,0.81D0)
4 A=C*B-A
  B=C*A-B
  QM=QQ*QM
  QN=QM*QN
  THETA1=QN*A+THETA1
  QM=QQ*QM
  QN=QM*QN
  THETA1=QN*B+THETA1
  IF(QN.GE.1.E-14) GO TO 4
  RETURN
END

```

C***** TRK *****88

```

SUBROUTINE TRK(M)
  IMPLICIT REAL*8 (A-H,O-Z)

```

```

  INCLUDE 'BINFF.CCC'
  INCLUDE 'BMI.CCC'
  INCLUDE 'DIM.CCC'
  INCLUDE 'TRKINT.CCC'
  INCLUDE 'CONTRL.CCC'
  INCLUDE 'CDERIV.CCC'
  INCLUDE 'CONST.CCC'
  INCLUDE 'FILES.CCC'
  common/seed/nsr

```

```

  DIMENSION V(7),V0(7),U(56),BX(9),BY(9),VV(7)
  LOGICAL DPRSAV
  DIMENSION R7(7,7),RW(3),U0(7),UP(7),UM(7),UF(7)
  EQUIVALENCE (AXCOS,V(1)), (AYCOS,V(3)), (DPP,U(6))
  INTEGER BDAT
  DIMENSION SCALE(4)
  DATA IBLNK,(RW(1),I=1,3)/1H ,3*0.0/

```

```

DATA DELP/1.D-08/
DATA IBLNK/4H /
C REVISED FEB. 1976
C INSTRUCTION TRK——TRACK PARTICLES THROUGH LINEAR OR
C NON-LINEAR TRANSFORMATIONS

C-----
C  T   TRK  NP IMP//V   BLIN/MAT  V0 /  NR IOF/IVSAV IMODE/IPR  IPL/
C                                     //XMAX   /DXMAX   /YMAX   /DYMAX
C-----

C NP=NUMBER PARTICLES.  THESE ARE STORED IN V, A PREVIOUS PVEC.
C NR = NUMBER OF TRANSITS OF BLIN TO BE MADE BY EACH PARTICLE.
C V = NAME OF PVEC INSTRUCTION CONTAINING INITIAL COORDINATES OF RAYS.
C V0 = NAME OF REFERENCE PARTICLE, ALSO DEFINED BY A PVEC.
C U = V0 + V IS THE PARTICLE VECTOR BEING TRACKED
C BLIN = NAME OF THE BEAM LINE, DEFINED BY A BML INSTRUCTION.
C IF BLANK, BEAM LINE CURRENTLY STORED IN MI ARRAY IS USED.
C MAT = NAME OF MATRIX THAT GIVES BETAS AT PRINT POSITIONS.
C IF MAT IS PRESENT, OUTPUT SHOWS THE ACTION VARIABLES EX, EY.
C IF MAT AND V0 ARE NAMED, OUTPUT SHOWS EX,EY AND PHASES PHIX,PHIY.
C IOF = OUTPUT FREQUENCY =
C           = 00001, EACH POSITION
C           = 0R00P, EVERY R'TH TRANSIT AT P'TH POSITION..

C IMP = INPUT OPTION
C IMP=0  V=X,DX,Y,DY,DS,DP/P (METERS,RADIANS)
C IMP=1  V=X,DX,Y,DY,DS,DP/P (MM,MRAD, DP/P IN 0/00)
C IMP=2  V=EX(MM-MRAD),PSIX(DEG),EY,PSIY
C REGARDLESS OF IMP, THE OUTPUT UNITS OF X,DX,Y,DY,DS,DP/P ARE M,RAD,M,RAD,M,1.
C THE OUTPUT UNITS OF EX,PHIX ARE MM-MRAD, DEGREES AND SIMILARLY FOR EY,PHIY.
C IF V0 IS LEFT BLANK, V STANDS FOR TOTAL VECTOR U, WITH 6 COMPONENTS
C U = X,DX,Y,DY,-DS,DP/P OR U = EX,PSIX,EY,PSIY,-DS,DP/P
C IVSAV = 1 SAVE TRACKED VECTORS. (REPLACE ORIGINAL VECTOR WITH
C NEW ONE IN ITS OWN STORAGE.)

C STORED VALUES WILL BE V AFTER TRACKING.
C STORED IN CARTESIAN COORDINATES, X,DX,Y,DY,...
C FOR SUBSEQUENT RUNS USING V, SET IMP = 0.
C OTHERWISE, INPUT VECTOR STORAGE REMAINS UNCHANGED.
C IMODE = 0,1 TRACK WITH STORED MATRICES AND MAPS
C           =2 TRACK USING REVMAT AND SAVE CUMULATIVE MATRIX IN
C STORAGE OF TRK.
C           =5 CALL REVMAT, DO NOT SAVE MATRIX
C (TRANSMITED TO TRKCN AS IOP)

C IPR = -1 NO PRINT
C IPR = 0 DO PRINT
C IPR = 2 PRINT IF TAGGED ONLY (4-8 PUNCH)
C IPL = -1,0 NO PLOT
C IPL = 1 PLOT
C
C XMAX,DXMAX,YMAX,DYMAX = SCALES FOR PHASE PLOTS (MM,MRAD)
C (CARD MUST ALWAYS BE INCLUDED)

```

```

DEGRAD = 1.D0/RADEG
NPARTS=INFF(4,M)
NRR = IDAT(M,1)
MVECS=MDAT(M,1)
NVECS=BDAT(M,1)
NBML=BDAT(M,2)
CALL DATA (M,1,1,4,SCALE)
NREF=2
MR = 4
NR = 1
IFLAG = 1
NV(1) = 1
NVEC0=BDAT(M,4)
IF (NVEC0.NE.IBLNK) GO TO 2
WRITE (3,1000) NBML
NREF=1
1 DO 3 J=1,7
  U(J)=0.
3 V0(J)=0.
  GO TO 4
2 MVEC0=MDAT(M,4)
  CALL DATA(MVEC0,1,1,7,V0)
  DO 21 J=1,5
21 U(J) = 1.D3*V0(J)
  U(6) = 100.D0*V0(6)
  U(7)=V0(7)
  WRITE (3,1006) NVEC0,NVEC0,(U(J),J=1,7)
4 CONTINUE
  IMP = INFF(5,M)
  IOF = IDAT(M,2)
  IVSAV = IDAT(M,3)
  IMODE = IDAT(M,4)
  IPR = IDAT(M,5)
  IPL = IDAT(M,6)
  IF (IMODE.EQ.0) IMODE = 1
  IF (IMODE.NE.2) GO TO 30
  NPARTS = 1
  CALL RTRV7(MUNIT,R7,RW)
  CALL STOR7(M,R7,RW)
30 IF (NBML.NE.IBLNK) GO TO 402
  NELS = IABS(NELS)
  GO TO 401
402 CONTINUE
  MBML=MDAT(M,2)
C PUT BEAM LIST IN MI ARRAY
  NDAT=INFF(17,MBML)
  CALL MIFILL(MBML,1,NDAT,NELS,MI)
  IF(ERROR) RETURN
C FIND BETAS FOR REFERENCE MATRIX MAT
401 NMAT=BDAT(M,3)
  IF (NMAT.EQ.IBLNK) GO TO 403
  MEL=MDAT(M,3)
  CALL BET(MEL,BX,BY,0)
403 CONTINUE
  PLT=.FALSE.

```



```

DPR=.FALSE.
IF (IPR.LT.0) DPR = .TRUE.
IF (IPL.GT.0) PLT = .TRUE.
KREV = IOF/1000
KPOS = MOD(IOF,1000)
IF (PLT) WRITE(TRK5) NPARTS,NRR,NVECS,NBML,NMAT,KPOS,NVEC0,
1      (U(J),J=1,7)

DO 250 IP=1,NPARTS
BEG=.TRUE.
IPS=IP*7-6
CALL DATA(MVECS,1,IPS,7,V)
IGO = 0
IF (IMODE.EQ.2) IGO = 1
10 IF (IMP.NE.2) GO TO 9
EX=V(1)*1.D-6
EY=V(3)*1.D-6
PHIX=V(2)*RADEG
PHIY=V(4)*RADEG
AX=SQRT(EX*BX(2))
AY=SQRT(EY*BY(2))
C=COS(PHIX)
S=SIN(PHIX)
V(1)=AX*C
V(2)=-AX*(S+BX(3)*C)/BX(2)
C=DCOS(PHIY)
S=DSIN(PHIY)
V(3)=AY*C
V(4)=-AY*(S+BY(3)*C)/BY(2)
9 IF (IMP.NE.1) GO TO 8
DO 11 I=1,6
11 V(I) = V(I)*1.D-3
8 IF (NREF.EQ.1) GO TO 81
DO 41 I=5,7
41 V(I) = 0.
81 DO 82 J=1,7
82 VV(J) = V(J)
83 DO 5 J=1,7
U(J) = V0(J) + V(J)
5 U0(J) = U(J)
C NM = INFF(2,M)
IF(IOF.EQ.0) IOF = 1
IF (NR.EQ.0) NR=1
C TRACK ONE PARTICLE THROUGH THE BEAM SYSTEM.
C PRINT HEADING AND INITIAL CONDITIONS
IF(DPR) GO TO 51
IF (IOF.NE.1.AND.NMAT.NE.IBLNK) GO TO 551
WRITE(3,1001)
GO TO 51
551 IF(NREF.EQ.1) WRITE(3,1021)
IF(NREF.EQ.2) WRITE(3,1011)
51 CONTINUE
IRV = 1
IPOS = 1
C FIND NAME OF PARTICLE

```

```

      IF(DPR) GO TO 52
      IF(NREF.EQ.2) WRITE(3,1012) NVECS,IP
      IF(NREF.EQ.1) WRITE(3,1002) NVECS,IP
52    CONTINUE
C    LOOP OVER REVOLUTIONS
55    DO 213 IRV = 1,NRR
      IRM=IRV-1
C      IF(.NOT.DPR.AND.IOF.EQ.1) WRITE(3,1005) IRM
C    LOOP THROUGH BEAM SYSTEM
      NELF=NELS
      IF (IRV.EQ.NRR) NELF = NELF + 1
C    RESET RANDOM SEED SO ERRORS REPEAT ON SUCCESSIVE TURNS
      nrr=nsr
      DO 212 IBS=1,NELF
        IPOS=IBS-1
C    IF INTEGRATION STEPS ARE BEING PRINTED, OMIT TRK PRINT.
63    IF (IOF.NE.01000) GO TO 62
        IF (IBS.NE.1) GO TO 211
62    DO 6 J=1,4
6      V(J) = 1.D3 * ( U(J) - V0(J) )
        IF(IOF.EQ.1) GO TO 214
61    IF (IPOS.NE.KPOS) GO TO 211
        IRVV = IRV - 1
        IF (KREV.EQ.0) GO TO 66
        IF (MOD(IRVV,KREV).NE.0) GO TO 211
66    IF (NMAT.EQ.IBLNK) GO TO 214
        WX = BX(2)*V(2)*V(2) + 2.D0*BX(3)*V(1)*V(2) + BX(4)*V(1)*V(1)
        WY = BY(2)*V(4)*V(4) + 2.D0*BY(3)*V(3)*V(4) + BY(4)*V(3)*V(3)
C    TIME FOR OUTPUT
        IF (NREF.NE.1) GO TO 7
        IF(DPR) GO TO 215
        WRITE (3,1005) IRM,IPOS,(U(I),I=1,6),WX,WY
        GO TO 215
7      CONTINUE
        IF (WX.NE.0.) GO TO 71
        PHIX = 0.
        GO TO 72
71     AXSIN = - BX(2)*V(2) - BX(3)*V(1)
        PHIX = DATAN2(AXSIN,AXCOS)*DEGRAD
72     IF (WY.NE.0.) GO TO 73
        PHIY = 0.
        GO TO 74
73     AYSIN = - BY(2)*V(4) - BY(3)*V(3)
        PHIY = DATAN2(AYSIN,AYCOS)*DEGRAD
74     CONTINUE
        IF(DPR) GO TO 215
        WRITE(3,1014) IRM,IPOS,DPP,WX,PHIX,WY,PHIY,(V(J),J=1,4)
        GO TO 215
214    CONTINUE
        IF (IPR.NE.2) GO TO 216
        MATI=INFF(2,NEL)
        DECODE (4,2000,MATI) MATL
2000  FORMAT (A1,3X)
        IF (MATL.NE.1H ) GO TO 215
216    CONTINUE

```

```

      IF (DPR) GO TO 215
      WRITE (3,1004) IRM,IPOS,(U(J),J=1,6)
215  CONTINUE
      IF (.NOT.PLT) GO TO 211
      WRITE(TRK5) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
      WRITE(TRK6) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
C ADVANCE THROUGH BEAM SYSTEM
211  CONTINUE
      IF (IBS.GT.NELS) GO TO 212
      NEL = MI(IBS)
      CALL TRKCN(M,NEL,U,0,IMODE)
212  CONTINUE
      nsr=nrn
213  CONTINUE
      IF (IGO.EQ.0) GO TO 245
      IF (IGO-2) 220,225,240
220  CALL RTRV7(M,R7,RW)
      DPRSAV=DPR
      DPR=.TRUE.
      DLP = -DELP
      DO 222 I=1,7
222  UF(I) = U(I)
      GO TO 230
225  DLP = + DELP
      DO 227 I=1,7
227  UM(I) = U(I)
230  DO 231 J=1,7
231  U(J) = V0(J) + VV(J)
      U(6) = U(6) + DLP
      IGO = IGO + 1
      GO TO 55
240  DO 242 I=1,7
242  UP(I) = U(I)
      DO 243 I=1,6
243  R7(I,6) = 0.5D0*(UP(I) - UM(I))/DELP
      CALL MXV7(R7,U0,U0)
      DO 244 I=1,5
244  R7(I,7) = UF(I) - U0(I)
      CALL STOR7(M,R7,RW)
      CALL MXV7(R7,V0,V0)
      DO 246 J=1,7
246  V(J) = UF(J) - V0(J)
      DPR=DPRSAV
      GO TO 248
245  CONTINUE
      DO 247 J=1,4
247  V(J) = U(J) - V0(J)
248  IF (IVSAV.EQ.1) CALL STDAT(MVECS,1,IPS,7,V)
250  CONTINUE

      CALL HED
      IF (.NOT.PLT) RETURN
      CALL TRKPLT(SCALE)
      RETURN
1000 FORMAT (27H PARTICLE TRACKING THROUGH A5,1H.)

```

```

1001 FORMAT (/ '    PARTICLE    REV.    POSITION          X          DX
      1          Y          DY          DS          DP/P ')
1002 FORMAT(4X,A5,I3)
1004 FORMAT (16X,I3,6X,I3,5X,6(1X,F12.8))
1005 FORMAT (16X,I3,6X,I3,5X,6F10.7,2F13.5)
1006 FORMAT (10X,'TRACKING RELATIVE TO REFERENCE RAY 'A5', ( UNITS(X,DX
      1,Y,DY,DS,DP/P) = (MM,MRAD,MM,MRAD,MM,0/0) )' /20X,A5' = '7F10.5)
1011 FORMAT(/ '    RAY    REV. POS.  DP/P  EX(MM-MR)    PHIX(DEG
      1) EY(MM-MR)    PHIY(DEG) X(MM)          DX(MR)          Y(MM)          DY(
      2MR)' )
1012 FORMAT (4X,A5,I2)
1014 FORMAT (10X,2I5,F11.3,8F12.5)
1021 FORMAT(/ '    PARTICLE    REV    POSITION'9X'X'8X'DX'9X'Y'8X'DY'
      16X'DS'6X'DP/P',7X,'EX(MM-MR)',4X,'EY(MM-MR)')
      END
C *****TRKB*****
C BETATRON FUNCTION TRACKING ROUTINE 9-19-85
C
      SUBROUTINE TRKB(M)
      IMPLICIT REAL*8 (A-H,O-Z)

      INCLUDE 'BSTORE.CCC'
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BMI1.CCC'

      DIMENSION MI(1)
      EQUIVALENCE (MI,MI1),(NELS,NELS1)
      INCLUDE 'CONTRL.CCC'
      INCLUDE 'MATCH3.CCC'
      INCLUDE 'CDERIV.CCC'

      DIMENSION BXI(9),BYI(9),V0(7),VI(7),VF(7),T(49)
      DIMENSION BX(6),BY(6),BX0(6),BY0(6),RX(2,3),RY(2,3),RW(3),BETF(14)
      DIMENSION P(8), V(7)
      EQUIVALENCE(BX,BETF),(BY,BETF(7)), (S,BETF(13)),(TH,BETF(14))
      EQUIVALENCE (V,VF(7))
      INTEGER BDAT,ELNAMEI,ELNAME
      EXTERNAL DER2,DER3,DER4,DELO
      LOGICAL NOSTOR,MPFLG,VECFLAG
      LOGICAL RADFLG
      DATA DZERO/0.0/
      DATA MP2FLG/0/

C
      DATA IBLANK/'    '/

C  INSTRUCTION TRKB——TRACK BETATRON FUNCTIONS

C  T  TRKB  IR1 IR2//BLIN BET0 V0          IT1 IT2 ITBL IRAD S0 TH0

C  RESERV  A TABLE FOR STORAGE OF BETA FUNCTIONS FROM POSITION IR1
C  THROUGH POSITION IR2 OF BET0.
C
C  TRACK BETA FUNCTIONS DEFINED BY [ BET0 IBET  //... ]
C  THROUGH BEAM LINE DEFINED BY [ BLIN BML  //... ] FROM IT1 TO IT2.
C

```

```

C TRACK A PARTICLE VECTOR DEFINED BY [ V0 PVEC //...]
C THROUGH THE BEAMLINE.
C
C BETA FUNCTION ARRAY IS STORED AT STORE(LQ3), LQ3=INFF(24,M),
C WHERE B(I,IPOS+1) = QX,BX,AX,GX,X,DX, QY,BY,AY,GY,Y,DY, S,THETAX.

C ITBL = 1 — INPUT BETAS FROM TRKB TABLE, AT IT1, EXCEPT
C ON 1ST CALL, AND/OR IF IT1=0, FROM IBET INPUT.
C BET0 = /NAME/ — INITIAL BETAS FROM ARRAY/NAME/, FOR AN IBET,
C AND FROM BETA-FUNCTIONS OF /NAME/, IF A MATRIX.
C IRAD = 1 — OUTPUT PHASE ADVANCES IN RADIANS
C = 0 — IN UNITS OF 2PI
C
C DEFAULTS —
C
C IF KA = KB = 0 (OR BLANK), A TABLE IS RESERVED SUFFICIENT TO STORE
C THE COMPLETE BEAM LINE.
C IF KA = -1 NO TABLE IS RESERVED, NO BETAS ARE STORED.
C
C IF KT1 = IT2 = 0 (OR BLANK), TRACKING IS DONE THROUGH COMPLETE BEAM-LINE.
C

```

```

RADFLG = .FALSE.
IF (MINFLG.NE.4) GO TO 23
IF (MIFLG) GO TO 152
GO TO 4
23 NOSTOR = .FALSE.
TPI=4.D0*DACOS(DZERO)
ELNAMEI=IBLANK
KB=INFF(5,M)
KA=INFF(4,M)
LQ3 = INFF(24,M)
NCOL = IDAT(M,5)
IF (KA.GT.-1) GO TO 151
NOSTOR = .TRUE.
GO TO 152
151 LTAB0 = LQ3 - NCOL*KA - 1
152 MBML = MDAT(M,1)

NDAT=INFF(17,MBML)
CALL MIFILL(MBML,1,NDAT,NELS,MI)
CALL MMM(M,NELS,MI)
IF (MINFLG.EQ.4) GO TO 4
IP1=IDAT(M,1)
IP2=IDAT(M,2)
ICALL=IDAT(M,6)
IRAD = IDAT(M,4)

SI = FDAT(M,1)
TI = FDAT(M,2)

```

```

C SET UP CONSTANTS FOR CALCULATING PSI IN DEGREES OR RADIANS
IF (IRAD.EQ.1) RADFLG=.TRUE.
DNOM = TPI
CCON = 1.D0

```

```

      IF (.NOT.RADFLG)   GO TO 153
      DNOM = 1.D0
      CCON = TPI
153  CONTINUE

C  GET VECTOR
      NAMVEC = BDAT(M,3)
      VECFLAG = .FALSE.
      IF(NAMVEC.EQ.1BLANK) GO TO 160
      VECFLAG = .TRUE.
      MVEC = MDAT(M,3)
      CALL DATA(MVEC,1,1,7,VI)
160  CONTINUE

      NAMBET=BDAT(M,2)
      NAME=INFF(2,M)

      ITABLE = IDAT(M,3)
      IF(ITABLE.EQ.1.AND..NOT.NOSTOR) GO TO 10
      GO TO 12

C 10  IF ((ICALL.EQ.0).OR.(IP1.EQ.0) ) GO TO 12
10  CONTINUE
C  INITIAL BETAS FROM BETA FUNCTION TABLE OF TRKB INSTRUCTION,
C  EXCEPT THAT FOR FIRST CALL, OR IF IP1=0, THEY ARE TAKEN FROM IBET OR MTR.
      LOCBET=NCOL*(IP1-KA)+1
      MBET=M
      NTYP=8
      CALL DATA(MBET,NTYP,LOCBET+12,1,SI)
      CALL DATA(MBET,NTYP,LOCBET+13,1,TI)
      IF(VECFLAG) CALL DATA(MBET,NTYP,LOCBET+14,7,VI)
      GO TO 13

C  INITIAL BETAS FROM ARRAY SPECIFIED BY NAMBET
12  LOCBET=1
      NTYP=1
      MBET=MDAT(M,2)

      NAMOP=INFF(1,MBET)

      IF(NAMOP.EQ.4HIBET) GO TO 13
C  INITIAL BETAS ARE BETAS OF MATRICES MBET.
      CALL BET(MBET,BXI,BYI,1)
      BXI(1)=0.
      BYI(1)=0.

      GO TO 16

13  CALL DATA(MBET,NTYP,LOCBET ,6,BXI)
      CALL DATA(MBET,NTYP,LOCBET+6,6,BYI)
C
16  CONTINUE

26  IF (IP1.EQ.0)  GO TO 9

```

```

      NELI = MI(IP1)
      ELNAMEI=MNAME(NELI)
9      IF(IP1.LT.KA) NOSTOR=.TRUE.
      IF(IP2.GT.KB) NOSTOR=.TRUE.
      ICALL=1
      CALL STDAT(M,3,3,1,ICALL)

4      CONTINUE

      S = SI
      TH = TI

      DO 3 I=1,6
      BX(I) = BXI(I)
      BY(I) = BYI(I)
      V(I) = VI(I)
3      CONTINUE
      V(7) = VI(7)

      IF (NOPR) GO TO 7
      CALL DASH
      LNUM = 0
      IF (I2.GT.45) WRITE (3,1004)
      IF(.NOT.VECFLAG) GO TO 406
      WRITE (3,1002)
      GO TO 407
406  WRITE(3,1000)
407  CONTINUE

      ELNAME=ELNAMEI
7      CONTINUE
      I1 = IP1 + 1
      I2 = IP2

      IF (BX(4).EQ.0.) BX(4)=(1.D0+BX(3)*BX(3))/BX(2)
      IF (BY(4).EQ.0.) BY(4)=(1.D0+BY(3)*BY(3))/BY(2)

C      LOOP OVER POSITIONS
19   DO 1 IBS = I1,I2
      IPOS = IBS - 1
      IF(NOPR) GO TO 6
      IF(.NOT.VECFLAG) GO TO 190
      IF (JMOD(LNUM,45).NE.0.OR.LNUM.EQ.0) GO TO 191
      WRITE (3,1004)
      WRITE (3,1002)
      LNUM = 0
191  IF (JMOD(LNUM,5).EQ.0.AND.LNUM.NE.0) WRITE (3,1010)
      WRITE(3,1003) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),
1      (BY(I),I=1,3),BY(5),BY(6),(V(I),I=1,4)
      LNUM = LNUM + 1
      GO TO 6
190  IF (JMOD(LNUM,45).NE.0.OR.LNUM.EQ.0) GO TO 192
      WRITE (3,1004)
      WRITE (3,1000)

```

```

      LNUM = 0
192  IF (JMOD(LNUM,5).EQ.0.AND.LNUM.NE.0) WRITE (3,1010)
      WRITE(3,1001) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),
1      (BY(I),I=1,3),BY(5),BY(6)
      LNUM = LNUM + 1
6    NEL = MI(IPS)
      IF(NOSTOR) GO TO 8
      J=LTAB0+NCOL*IP0S
      DO 11 I=1,14
11   STORE(J+I)=BETF(I)
      IF(.NOT.VECFLAG) GO TO 8
      DO 110 I=15,21
110  STORE(J+I) = VF(I)
8    CONTINUE

      ELNAME=MNAME(NEL)
      STOT = S
C
18   CALL RXY(NEL,RX,RY,RW)

      IF(.NOT.VECFLAG) GO TO 180
      DO 181 I=1,6
      BX0(I)=BX(I)
      BY0(I)=BY(I)
181  V0(I) = V(I)
      V0(7) = V(7)
      GO TO 182

180  DO 2 I=1,6
      BX0(I)=BX(I)
2    BY0(I)=BY(I)

182  S = S + RW(1)
      TH = TH + RW(2)

      BX(2)=RX(1,1)*RX(1,1)*BX0(2)-2.D0*RX(1,1)*RX(1,2)*BX0(3)
1      + RX(1,2)*RX(1,2)*BX0(4)

      BY(2)=RY(1,1)*RY(1,1)*BY0(2)-2.D0*RY(1,1)*RY(1,2)*BY0(3)
1      + RY(1,2)*RY(1,2)*BY0(4)

      BX(3) = -RX(1,1)*RX(2,1)*BX0(2)
1      + (RX(1,1)*RX(2,2)+RX(1,2)*RX(2,1))*BX0(3)
2      - RX(1,2)*RX(2,2)*BX0(4)

      BY(3) = -RY(1,1)*RY(2,1)*BY0(2)
1      + (RY(1,1)*RY(2,2)+RY(1,2)*RY(2,1))*BY0(3)
2      - RY(1,2)*RY(2,2)*BY0(4)

      BX(4) = RX(2,1)*RX(2,1)*BX0(2) - 2.D0*RX(2,1)*RX(2,2)*BX0(3)
1      + RX(2,2)*RX(2,2)*BX0(4)

      BY(4) = RY(2,1)*RY(2,1)*BY0(2) - 2.D0*RY(2,1)*RY(2,2)*BY0(3)
1      + RY(2,2)*RY(2,2)*BY0(4)

```


BX(5) = RX(1,1)*BX0(5) + RX(1,2)*BX0(6) + RX(1,3)

BY(5) = RY(1,1)*BY0(5) + RY(1,2)*BY0(6) + RY(1,3)

BX(6) = RX(2,1)*BX0(5) + RX(2,2)*BX0(6) + RX(2,3)

BY(6) = RY(2,1)*BY0(5) + RY(2,2)*BY0(6) + RY(2,3)

IF(NONU) GO TO 183

BETAV=DSQRT(BX(2)*BX0(2))

BERAT=DSQRT(BX0(2)/BX(2))

SINE=RX(1,2)/BETAV

COSINE=RX(1,1)*BERAT-BX0(3)*SINE

C DNOM OR CCON SET TO TPI OR 1. DEPENDING ON DEGREES OR RADIANS.

DNU = DATAN2(SINE,COSINE)/DNOM

IF (DNU.LT.0) DNU = DNU + CCON

BX(1) = BX0(1) + DNU

BETAV=DSQRT(BY(2)*BY0(2))

BERAT=DSQRT(BY0(2)/BY(2))

SINE=RY(1,2)/BETAV

COSINE=RY(1,1)*BERAT-BY0(3)*SINE

DNU = DATAN2(SINE,COSINE)/DNOM

IF (DNU.LT.0) DNU = DNU + CCON

BY(1) = BY0(1) + DNU

183 CONTINUE

IF(.NOT.VECFLAG) GO TO 1

CALL RTRV7(NEL,T,RW)

CALL MXV7(T.V0,V)

1 CONTINUE

17 IPOS = I2

MP2FLG = 0

IF(NOSTOR) GO TO 5

J = J + NCOL

DO 14 I=1,14

14 STORE(J+I)=BETF(I)

IF(.NOT.VECFLAG) GO TO 5

DO 140 I=15,21

140 STORE(J+I) = VF(I)

5 CONTINUE

IF(NOPR) RETURN

IF(.NOT.VECFLAG) GO TO 150

WRITE(3,1003) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),

1 (BY(I),I=1,3),BY(5),BY(6),(V(I),I=1,4)

GO TO 154

150 WRITE(3,1001) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),

1 (BY(I), I=1,3), BY(5), BY(6)

154 CALL DASH

RETURN

1000 FORMAT (' POS', 11X, 'S', 7X, 'QX', 10X, 'BX', 8X, 'AX', 8X, 'X', 9X,
1 'DX', 14X, 'QY', 10X, 'BY', 8X, 'AY', 8X, 'Y', 9X, 'DY')

1001 FORMAT (I4, 1X, A4, 1X, F10.4, 2(F10.6, F11.4, F11.6, F10.6, F10.6, 6X))

1002 FORMAT (' POS', 10X, 'S', 7X, 'QX', 7X, 'BX', 7X, 'AX', 6X, 'DX', 6X,
1 'DDX', 6X, 'QY', 7X, 'BY', 7X, 'AY', 6X, 'DY', 6X, 'DDY', 4X, 'X(MM)',
2 1X, 'DX(MR)', 1X, 'Y(MM)', 1X, 'DY(MR)')

1003 FORMAT (I4, 1X, A4, F10.4, 2(F8.4, 2F9.4, 2F8.4, 1X), 2(F6.2, F7.3))

1004 FORMAT (1H1)

1010 FORMAT (1H)

END

C *****TRKE*****

C ENVELOPE TRACKING ROUTINE 9-19-85

C

SUBROUTINE TRKE(M)
IMPLICIT REAL*8 (A-H, O-Z)

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BMI1.CCC'

DIMENSION MI(1)
EQUIVALENCE (MI, MI1), (NELS, NELS1)
INCLUDE 'CONTRL.CCC'
INCLUDE 'MATCH3.CCC'
INCLUDE 'CDERIV.CCC'

DIMENSION BXI(9), BYI(9), V0(7), VI(7), VF(7), T(49)
DIMENSION BX(6), BY(6), BX0(6), BY0(6), RX(2,3), RY(2,3), RW(3), BETF(14)
DIMENSION P(8), V(7)
EQUIVALENCE (BX, BETF), (BY, BETF(7)), (S, BETF(13)), (TH, BETF(14))
EQUIVALENCE (EPX, P(1)), (EPY, P(2)), (EPL, P(3)), (DPP, P(4)), (V, VF(7))
INTEGER BDAT, ELNAMEI, ELNAME
EXTERNAL DER2, DER3, DER4, DELQ
LOGICAL NOSTOR, MPFLG, VECFLAG
LOGICAL RADFLG
DATA DZERO/0.0/
DATA MP2FLG/0/

C

DATA IBLANK/' '/

C INSTRUCTION TRKE——TRACK BEAM ENVELOPES

C T TRKE IR1 IR2//BLIN BET0 V0 EPS IT1 IT2 ITBL IADD EXCO EYCO

C RESERV A TABLE FOR STORAGE OF BETA FUNCTIONS FROM POSITION IR1

```

C THROUGH POSITION IR2 OF BET0.
C
C TRACK BETA FUNCTIONS DEFINED BY [ BET0 IBET  //... ]
C THROUGH BEAM LINE DEFINED BY [ B1 IN BML  //... ] FROM IT1 TO IT2,
C CALCULATE AND DISPLAY BEAM ENVELOPES.
C THE EMITTANCES ARE DEFINED BY [ EPS BVAL  //... ].
C
C TRACK A PARTICLE VECTOR DEFINED BY [ V0 PVEC  //... ]
C THROUGH THE BEAMLINE.
C
C BETA FUNCTION ARRAY IS STORED AT STORE(LQ3), LQ3=INFF(24,M),
C WHERE B(I,IPOS+1) = QX,BX,AX,GX,X,DX, QY,BY,AY,GY,Y,DY, S,THETAX.

C ITBL = 1 — INPUT BETAS FROM TRKB TABLE, AT IT1, EXCEPT
C ON 1ST CALL, AND/OR IF IT1=0, FROM IBET INPUT.
C BET0 = /NAME/ — INITIAL BETAS FROM ARRAY/NAME/, FOR AN IBET,
C AND FROM BETA-FUNCTIONS OF /NAME/, IF A MATRIX.
C IADD = 0 — ADD XB AND XP IN QUADRATURE, ETC.
C IADD = 1 — ADD XB AND XP ALGEBRAICALLY, ETC.
C
C DEFAULTS —
C
C IF KA = KB = 0 (OR BLANK), A TABLE IS RESERVED SUFFICIENT TO STORE
C THE COMPLETE BEAM LINE.
C IF KA = -1 NO TABLE IS RESERVED, NO BETAS ARE STORED.
C
C IF KT1 = IT2 = 0 (OR BLANK), TRACKING IS DONE THROUGH COMPLETE BEAM-LINE.
C

RADFLG = .FALSE.
IF (MINFLG.NE.4) GO TO 23
IF (MIFLG) GO TO 152
GO TO 4
23 NOSTCR = .FALSE.
TPI=4.D0*DACOS(DZERO)
ELNAMEI=IBLANK
KB=INFF(5,M)
KA=INFF(4,M)
LQ3 = INFF(24,M)
NCOL = IDAT(M,5)
IF (KA.GT.-1) GO TO 151
NOSTOR = .TRUE.
GO TO 152
151 LTAB0 = LQ3 - NCOL*KA - 1
152 MBML = MDAT(M,1)

NDAT=INFF(17,MBML)
CALL MIFILL(MBML,1,NDAT,NELS,MI)
CALL MMM(M,NELS,MI)
IF (MINFLG.EQ.4) GO TO 4
IP1=IDAT(M,1)
IP2=IDAT(M,2)
ICALL=IDAT(M,6)
IADD = IDAT(M,4)
SI = 0.0

```

```

TI = 0.0
IG = 1
IF(IADD.EQ.0) IG = 2
DNOM = TPI
CCON = 1.D0

```

C GET VECTOR

```

NAMVEC = BDAT(M,3)
VECFLAG = .FALSE.
IF(NAMVEC.EQ.IBLANK) GO TO 160
VECFLAG = .TRUE.
MVEC = MDAT(M,3)
CALL DATA(MVEC,1,1,7,VI)

```

160 CONTINUE

```

MBVAL = MDAT(M,4)
CALL DATA(MBVAL,5,1,4,P)
NAMBET=BDAT(M,2)
NAME=INFF(2,M)

ITABLE = IDAT(M,3)
IF(ITABLE.EQ.1.AND..NOT.NOSTOR) GO TO 10
GO TO 12

```

C 10 IF ((ICALL.EQ.0).OR.(IP1.EQ.0)) GO TO 12

10 CONTINUE

C INITIAL BETAS FROM BETA FUNCTION TABLE OF TRKB INSTRUCTION,

C EXCEPT THAT FOR FIRST CALL, OR IF IP1=0, THEY ARE TAKEN FROM IBET OR MTR.

```

LOCBET=NCOL*(IP1-KA)+1
MBET=M
NTYP=8
CALL DATA(MBET,NTYP,LOCBET+12,1,SI)
CALL DATA(MBET,NTYP,LOCBET+13,1,TI)
IF(VECFLAG) CALL DATA(MBET,NTYP,LOCBET+14,7,VI)
GO TO 13

```

C INITIAL BETAS FROM ARRAY SPECIFIED BY NAMBET

12 LOCBET=1

NTYP=1

MBET=MDAT(M,2)

NAMOP=INFF(1,MBET)

IF(NAMOP.EQ.4HIBET) GO TO 13

C INITIAL BETAS ARE BETAS OF MATRICES MBET.

CALL BET(MBET,BXI,BYI,1)

BXI(1)=0.

BYI(1)=0.

GO TO 16

13 CALL DATA(MBET,NTYP,LOCBET ,6,BXI)

CALL DATA(MBET,NTYP,LOCBET+6,6,BYI)

C

16 CONTINUE

C GET EMITTANCES

```

EPXCO = FDAT(M,1)
EPYCO = FDAT(M,2)
RATX = DSQRT(EPXCO/EPX)
RATY = DSQRT(EPYCO/EPY)

```

```

26 IF (IP1.EQ.0) GO TO 9
NELI = MI(IP1)
ELNAMEI=MNAME(NELI)
9 IF (IP1.LT.KA) NOSTOR=.TRUE.
IF (IP2.GT.KB) NOSTOR=.TRUE.
ICALL=1
CALL STDAT(M,3,3,1,ICALL)

```

4 CONTINUE

```

S = SI
TH = TI

```

```

DO 3 I=1,6
BX(I) = BXI(I)
BY(I) = BYI(I)
V(I) = VI(I)

```

```

3 CONTINUE
V(7) = VI(7)

```

```

I1 = IP1 + 1
I2 = IP2
IF (NOPR) GO TO 7
LNUM = 0
IF (I2.GT.40) WRITE (3,4004)

```

```

CALL DASH
WRITE(3,4001) P(1),EPXCO,P(3),P(2),EPYCO,P(4)
GO TO (404,405) IG

```

```

404 WRITE (3,4006)
GO TO 406

```

```

405 WRITE (3,4008)
IF(.NOT.VECFLAG) GO TO 406
WRITE (3,4011)
GO TO 407

```

```

406 WRITE(3,4010)

```

```

407 CONTINUE

```

```

ELNAME=ELNAMEI

```

7 CONTINUE

```

IF (BX(4).EQ.0.) BX(4)=(1.D0+BX(3)*BX(3))/BX(2)
IF (BY(4).EQ.0.) BY(4)=(1.D0+BY(3)*BY(3))/BY(2)

```

C COMPUTE AND PRINT BEAM ENVELOPES

```

X=DSQRT(EPX*BX(2))
Y=DSQRT(EPY*BY(2))

```

```

XP=BX(5)*DPP
YP=BY(5)*DPP
XCO=RATX*X
YCO=RATY*Y
XPR=DSQRT(EPX*BX(4))
YPR=DSQRT(EPY*BY(4))
XPPR=BX(6)*DPP
YPPR=BY(6)*DPP
XCOPR=RATX*XPR
YCOPR=RATY*YPR
GO TO (4116,4117) IG
4116 XTOT = X + DABS(XP)
      XPRTOT = XPR + DABS(XPPR)
      YTOT = Y + DABS(YP)
      YPRTOT = YPR + DABS(YPPR)
      GO TO 4118
4117 CONTINUE
      XTOT = DSQRT( X * X + XP * XP )
      XPRTOT = SQRT( XPR*XPR + XPPR*XPPR)
      YTOT = DSQRT(Y*Y + YP*YP)
      YPRTOT = DSQRT(YPR*YPR + YPPR*YPPR)
4118 CONTINUE
      XTOT=XTOT+XCO
      YTOT=YTOT+YCO
      XPRTOT=XPRTOT+XCOPR
      YPRTOT=YPRTOT+YCOPR

C  LOOP OVER POSITIONS
19  DO 1 IBS = I1,I2
      IPOS = IBS - 1
      IF(NOPR) GO TO 6
      IF(.NOT.VECFLAG) GO TO 190
      IF (JMOD(LNUM,40).NE.0.OR.LNUM.EQ.0) GO TO 191
      WRITE (3,4004)
      WRITE (3,4011)
      LNUM = 0
191  IF (JMOD(LNUM,40).NE.0.OR.LNUM.NE.0) WRITE (3,4040)
      WRITE(3,4012) IPOS,ELNAME,S,X,XP,XCO,XTOT,Y,YP,YCO,YTOT,
1      XPRTOT,YPRTOT,(V(I),I=1,4)
      LNUM = LNUM + 1
      GO TO 6
190  IF (JMOD(LNUM,40).NE.0.OR.LNUM.EQ.0) GO TO 192
      WRITE (3,4004)
      WRITE (3,4010)
      LNUM = 0
192  IF (JMOD(LNUM,5).EQ.0.AND.LNUM.NE.0) WRITE (3,4040)
      WRITE(3,4012) IPOS,ELNAME,S,X,XP,XCO,XTOT,Y,YP,YCO,YTOT,
1      XPR,XPPR,XPRTOT,YPR,YPPR,YPRTOT
      LNUM = LNUM + 1
6    NEL = MI(IFS)
      IF(NOSTOR) GO TO 8
      J=LTAB0+NCOL*IPOS
      DO 11 I=1,14
11   STCRE(J+I)=BETF(I)
      IF(.NOT.VECFLAG) GO TO 8

```

```

      DO 110 I=15,21
110  STORE(J+I) = VF(I)
      B   CONTINUE

```

```

      ELNAME=MNAME(NEL)
      STOT = S

```

C

```

18  CALL RXY(NEL,RX,RY,RW)

```

```

      IF(.NOT.VECFLAG) GO TO 180

```

```

      DO 181 I=1,6

```

```

      BX0(I)=BX(I)

```

```

      BY0(I)=BY(I)

```

```

181  V0(I) = V(I)

```

```

      V0(7) = V(7)

```

```

      GO TO 182

```

```

180  DO 2 I=1,6

```

```

      BX0(I)=BX(I)

```

```

2    BY0(I)=BY(I)

```

```

182  S = S + RW(1)

```

```

      TH = TH + RW(2)

```

```

      BX(2)=RX(1,1)*RX(1,1)*BX0(2)-2.D0*RX(1,1)*RX(1,2)*BX0(3)
1      + RX(1,2)*RX(1,2)*BX0(4)

```

```

      BY(2)=RY(1,1)*RY(1,1)*BY0(2)-2.D0*RY(1,1)*RY(1,2)*BY0(3)
1      + RY(1,2)*RY(1,2)*BY0(4)

```

```

      BX(3) = -RX(1,1)*RX(2,1)*BX0(2)
1      + (RX(1,1)*RX(2,2)+RX(1,2)*RX(2,1))*BX0(3)
2      - RX(1,2)*RX(2,2)*BX0(4)

```

```

      BY(3) = -RY(1,1)*RY(2,1)*BY0(2)
1      + (RY(1,1)*RY(2,2)+RY(1,2)*RY(2,1))*BY0(3)
2      - RY(1,2)*RY(2,2)*BY0(4)

```

```

      BX(4) = RX(2,1)*RX(2,1)*BX0(2) - 2.D0*RX(2,1)*RX(2,2)*BX0(3)
1      + RX(2,2)*RX(2,2)*BX0(4)

```

```

      BY(4) = RY(2,1)*RY(2,1)*BY0(2) - 2.D0*RY(2,1)*RY(2,2)*BY0(3)
1      + RY(2,2)*RY(2,2)*BY0(4)

```

```

      BX(5) = RX(1,1)*BX0(5) + RX(1,2)*BX0(6) + RX(1,3)

```

```

      BY(5) = RY(1,1)*BY0(5) + RY(1,2)*BY0(6) + RY(1,3)

```

```

      BX(6) = RX(2,1)*BX0(5) + RX(2,2)*BX0(6) + RX(2,3)

```

```

      BY(6) = RY(2,1)*BY0(5) + RY(2,2)*BY0(6) + RY(2,3)

```

```

      IF(NONU) GO TO 183

```

```

      BETAV=DSQRT(BX(2)*BX0(2))

```

```

BERAT=DSQRT(BX0(2)/BX(2))
SINE=RX(1,2)/BETAV
COSINE=RX(1,1)*BERAT-BX0(3)*SINE
C  DNOM OR CCON SET TO TPI OR 1. DEPENDING ON DEGREES OR RADIANS.
   DNU = DATAN2(SINE,COSINE)/DNOM
   IF (DNU.LT.0) DNU = DNU + CCON
   BX(1) = BX0(1) + DNU
   BETAV=DSQRT(BY(2)*BY0(2))
   BERAT=DSQRT(BY0(2)/BY(2))
   SINE=RY(1,2)/BETAV
   COSINE=RY(1,1)*BERAT-BY0(3)*SINE
   DNU = DATAN2(SINE,COSINE)/DNOM
   IF (DNU.LT.0) DNU = DNU + CCON
   BY(1) = BY0(1) + DNU

183  CONTINUE

   IF(.NOT.VECFLAG) GO TO 184
   CALL RTRV7(NEL,T,RW)
   CALL MXV7(T,V0,V)

184  CONTINUE

C      COMPUTE AND PRINT BEAM ENVELOPES
X=DSQRT(EPX*BX(2))
Y=DSQRT(EPY*BY(2))
XP=BX(5)*DPP
YP=BY(5)*DPP
XCO=RATX*X
YCO=RATY*Y
XPR=DSQRT(EPX*BX(4))
YPR=DSQRT(EPY*BY(4))
XPPR=BX(6)*DPP
YPPR=BY(6)*DPP
XCOPR=RATX*XPR
YCOPR=RATY*YPR
GO TO (5116,5117) IG
5116 XTOT = X + DABS(XP)
   XPRTOT = XPR + DABS(XPPR)
   YTOT = Y + DABS(YP)
   YPRTOT = YPR + DABS(YPPR)
   GO TO 5118
5117 CONTINUE
   XTOT = DSQRT( X * X + XP * XP )
   XPRTOT = SQRT( XPR*XPR + XPPR*XPPR)
   YTOT = DSQRT(Y*Y + YP*YP)
   YPRTOT = DSQRT(YPR*YPR + YPPR*YPPR)
5118 CONTINUE
   XTOT=XTOT+XCO
   YTOT=YTOT+YCO
   XPRTOT=XPRTOT+XCOPR
   YPRTOT=YPRTOT+YCOPR

1  CONTINUE

```



```

17  IPOS = 12
    MP2FLG = 0
    IF(NOSTOR) GO TO 5

    J = J + NCOL
    DO 14 I=1,14
14  STORE(J+I)=BETF(I)
    IF(.NOT.VECFLAG) GO TO 5
    DO 140 I=15,21
140 STORE(J+I) = VF(I)

5   CONTINUE

    IF(NOPR) RETURN

    IF(.NOT.VECFLAG) GO TO 150

    WRITE(3,4012) IPOS,ELNAME,S,X,XP,XCO,XTOT,Y,YP,YCO,YTOT,
1    XPRTOT,YPRTOT,(V(I),I=1,4)
    GO TO 153
150 WRITE(3,4012) IPOS,ELNAME,S,X,XP,XCO,XTOT,Y,YP,YCO,YTOT,
1    XPR,XPPR,XPRTOT,YPR,YPPR,YPRTOT
153 CALL DASH

    RETURN

4001  FORMAT (3X,'BEAM ENVELOPES (MM,MRAD)'/
1     5X,'EMITTANCES (MM-MRAD) — EPSX = ',F10.6,5X,'EPSXCO = ',
2     F10.6,5X,'EPSL = ',F13.6/30X,'EPSY = ',F10.6,5X,'EPSYCO = ',
3     F10.6,5X,'SIGP = ',F10.6,' (0/00)')

4002 FORMAT(1H+,90X,'DISPLACEMENT = ',F5.2,1H*,'SIGMA')
4006 FORMAT (5X,'SIGX = XB + XP + XCO,   ETC. '/')
4008 FORMAT (5X,'SIGX = SQRT(XB*XB + XP*XP) + XCO,   ETC. '/')

4010 FORMAT(' POS',10X,'S',6X,'XB',6X,'XP',5X,'XCO',4X,'SIGX',3X,
1         'YB',6X,'YP',5X,'YCO',4X,'SIGY',4X,
2         'DXB',5X,'DXP',5X,'DSIGX',3X,'DYB',5X,'DYP',5X,'DSIGY')

4011 FORMAT(' POS',10X,'S',6X,'XB',6X,'XP',5X,'XCO',4X,'SIGX',3X,
1         'YB',6X,'YP',5X,'YCO',4X,'SIGY',4X,'DSIGX',3X,'DSIGY',
2         3X,'XCEN',4X,'DXCEN',3X,'YCEN',4X,'DYCEN')

4012 FORMAT(I4,1X,A4,F10.4,2(F7.3,F8.3,2F7.3),6F8.3)
4004  FORMAT (1H1)
4040  FORMAT (1H )
    END

C *****TRKM*****
C MAP-TYPE ELEMENT TRACKING ROUTINE 9-19-85
C
    SUBROUTINE TRKM(M)
    IMPLICIT REAL*8 (A-H,O-Z)

```

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BMI1.CCC'

```

```

DIMENSION MI(1)
EQUIVALENCE (MI,MI1),(NELS,NELS1)
INCLUDE 'CONTRL.CCC'
INCLUDE 'MATCH3.CCC'
INCLUDE 'CDERIV.CCC'

```

```

DIMENSION BXI(9),BYI(9),V0(6),VI(6),VF(7),T(49)
DIMENSION BX(6),BY(6),BX0(6),BY0(6),RX(2,3),RY(2,3),RW(3),BETF(14)
DIMENSION P(8), V(7)
EQUIVALENCE(BX,BETF),(BY,BETF(7)), (S,BETF(13)),(TH,BETF(14))
EQUIVALENCE (V,VF(7))
INTEGER BDAT,ELNAMEI,ELNAME
EXTERNAL DER2,DER3,DER4,DELQ
LOGICAL NOSTOR,MPFLG,VECFLAG
LOGICAL RADFLG
DATA DZERO/0.0/
DATA MP2FLG/0/

```

```

C
DATA IBLANK/' '

```

```

C INSTRUCTION TRKM——TRACK THROUGH ELEMENTS DEFINED BY MAP STATEMENTS

```

```

C T TRKM IR1 IR2//BLIN BET0 MAPK PAR IT1 IT2 ITBL IRAD HSTEP

```

```

C RESERV A TABLE FOR STORAGE OF BETA FUNCTIONS FROM POSITION IR1
C THROUGH POSITION IR2 OF BET0.

```

```

C
C TRACK BETA FUNCTIONS DEFINED BY [ BET0 IBET //... ]
C THROUGH BEAM LINE DEFINED BY [ BLIN BML //... ] FROM IT1 TO IT2.

```

```

C
C TRACK A PARTICLE VECTOR DEFINED BY [ V0 PVEC //... ]
C THROUGH THE BEAMLINE.

```

```

C
C BETA FUNCTION ARRAY IS STORED AT STORE(LQ3), LQ3=INFF(24,M),
C WHERE B(I,IPOS+1) = OX,BX,AX,GX,X,DX, QY,BY,AY,GY,Y,DY, S,THETAX.

```

```

C ITBL = 1 — INPUT BETAS FROM TRKB TABLE, AT IT1, EXCEPT
C ON 1ST CALL, AND/OR IF IT1=0, FROM IBET INPUT.
C BET0 = /NAME/ —INITIAL BETAS FROM ARRAY/NAME/,FOR AN IBET,
C AND FROM BETA-FUNCTIONS OF /NAME/, IF A MATRIX.
C IRAD = 1 — OUTPUT PHASE ADVANCES IN RADIANS
C = 0 — IN UNITS OF 2PI

```

```

C
C DEFAULTS —

```

```

C
C IF KA = KB = 0 (OR BLANK), A TABLE IS RESERVED SUFFICIENT TO STORE
C THE COMPLETE BEAM LINE.
C IF KA = -1 NO TABLE IS RESERVED, NO BETAS ARE STORED.
C
C IF KT1 = IT2 = 0 (OR BLANK), TRACKING IS DONE THROUGH COMPLETE BEAM-LINE.
C

```

```

RADFLG = .FALSE.
IF (MINFLG.NE.4) GO TO 23
IF (MIFLG) GO TO 152
GO TO 4
23  NOSTOR = .FALSE.
    TFI=4.D0*DACOS(DZERO)
    ELNAMEI=IBLANK
    KB=INFF(5,M)
    KA=INFF(4,M)
    LQ3 = INFF(24,M)
    NCOL = IDAT(M,5)
    IF (KA.GT.-1) GO TO 151
    NOSTOR = .TRUE.
    GO TO 152
151  LTAB0 = LQ3 - NCOL*KA - 1
152  MBML = MDAT(M,1)

    NDAT=INFF(17,MBML)
    CALL MIFILL(MBML,1,NDAT,NELS,MI)
    CALL MAM(M,NELS,MI)
    IF (MINFLG.EQ.4) GO TO 4
    IP1=IDAT(M,1)
    IP2=IDAT(M,2)
    ICALL=IDAT(M,6)
    SI = FDAT(M,2)
    TI = 0.
    H0 = FDAT(M,1)
    DAT2(7) = H0
    MAPK = BDAT(M,3)
    IRAD = IDAT(M,4)
C   SET UP CONSTANTS FOR CALCULATING PSI IN DEGREES OR RADIANS
    IF (IRAD.EQ.1) RADFLG=.TRUE.
    DNOM = TPI
    CCON = 1.D0
    IF (.NOT.RADFLG) GO TO 153
    DNOM = 1.D0
    CCON = TPI
153  CONTINUE

    NAMBET=BDAT(M,2)
    NAMVEC = BDAT(M,3)
    NAMPAR = BDAT(M,4)
    NAME=INFF(2,M)

    ITABLE = IDAT(M,3)
    IF(ITABLE.EQ.1.AND..NOT.NOSTOR) GO TO 10
    GO TO 12

C 10  IF ((ICALL.EQ.0).OR.(IP1.EQ.0) ) GO TO 12
    10  CONTINUE
C  INITIAL BETAS FROM BETA FUNCTION TABLE OF TRKB INSTRUCTION.
C  EXCEPT THAT FOR FIRST CALL, OR IF IP1=0, THEY ARE TAKEN FROM IBET OR MTR.
    LOCBET=NCOL*(IP1-KA)+1
    MBET=M

```

```

NTYP=8
CALL DATA(MBET,NTYP,LOCBET+12,1,SI)
CALL DATA(MBET,NTYP,LOCBET+13,1,TI)
GO TO 13

```

C INITIAL BETAS FROM ARRAY SPECIFIED BY NAMBET

```

12  LOCBET=1
    NTYP=1
    MBET=MDAT(M,2)

```

```

    NAMOP=INFF(1,MBET)

```

```

    IF(NAMOP.EQ.4HIBET) GO TO 13

```

C INITIAL BETAS ARE BETAS OF MATRICES MBET.

```

    CALL BET(MBET,BXI,BYI,1)
    BXI(1)=0.
    BYI(1)=0.

```

```

    GO TO 16

```

```

13  CALL DATA(MBET,NTYP,LOCBET ,6,BXI)
    CALL DATA(MBET,NTYP,LOCBET+6,6,BYI)

```

```

16  CONTINUE

```

```

    IF (MAPK.EQ.4HMAP2) MP2FLG=1
    IF (MAPK.NE.3HELQ) GO TO 34
    NUM = 7
    GO TO 35

```

```

34  DECODE(10,1002,MAPK) NUM
1002 FORMAT(3X,I1)

```

```

35  IF (NAMPAR.EQ.IBLANK) GO TO 26
    MPAR = MDAT(M,4)
    CALL DATA(MPAR,1,1,6,DAT2)

```

```

26  IF (IP1.EQ.0) GO TO 9
    NELI = MI(IP1)
    ELNAMEI=MNAME(NELI)

```

```

9   IF(IP1.LT.KA) NOSTOR=.TRUE.
    IF(IP2.GT.KB) NOSTOR=.TRUE.
    ICALL=1
    CALL STDAT(M,3,3,1,ICALL)

```

```

4   CONTINUE
    S = SI
    TH = TI

```

```

    DO 3 I=1,6
    BX(I) = BXI(I)
    BY(I) = BYI(I)

```

```

3   CONTINUE
    I1 = IP1 + 1
    I2 = IP2

```

```

      IF (NOPR) GO TO 7
      IF (I2.GT.45) WRITE (3,1004)
      LNUM = 0
      CALL DASH
406  WRITE(3,1000)

```

```

      ELNAME=ELNAMEI
7    CONTINUE

```

C ALL ELEMENTS ARE DEFINED BY MAPS

```

21  NR = 1
      IFLAG = 1
      NV(1) = 1
      S = 0.
      TH = 0.
      IF (NUM-3) 27,28,29

```

C MAP2

```

27  MR = 8
      GO TO 19

```

C MAP3 — TRANSVERSE AND LONGITUDINAL ENVELOPES

```

28  MR = 6
      GO TO 19

```

C MAP4 — TRANSVERSE ENVELOPES

```

29  MR = 4

```

C LOOP OVER POSITIONS

```

19  DO 1 IBS = I1,I2
      IPOS = IBS - 1
      IF(NOPR) GO TO 6

```

```

190 IF (JMOD(LNUM,45).NE.0.OR.LNUM.EQ.0) GO TO 191
      WRITE (3,1004)
      WRITE (3,1000)
      LNUM = 0

```

```

191 IF (JMOD(LNUM,5).EQ.0.AND.LNUM.NE.0) WRITE (3,1003)
      WRITE(3,1001) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),
1      (BY(I),I=1,3),BY(5),BY(6)
      LNUM = LNUM + 1

```

```

6    NEL = MI(IFS)
      IF(NOSTOR) GO TO 8
      J=LTAB0+NCOL+IPOS
      DO 11 I=1,14

```

```

11  STORE(J+I)=BETF(I)
8    CONTINUE

```

```

      ELNAME=MNAME(NEL)
      STOT = S

```

C

```

      MAPK2=INFF(1,NEL)
      IF (MAPK2.EQ.MAPK) GO TO 20
      MAPK=MAPK2
      IF (MAPK.EQ.3HELQ) GO TO 22

```

```

      NUM=4
      GO TO 32
22  NUM=7
      GO TO 36
20  IF (NUM.EQ.7) GO TO 36
      IF (NUM-3) 30,31,32

```

C MAP2

```

30  P(1) = BX(2)
      P(2) = BX(3)
      P(3) = BX(5)
      P(4) = BX(6)
      P(5) = BY(2)
      P(6) = BY(3)
      P(7) = BY(5)
      P(8) = BY(6)
      CALL DIFEQ(NEL,P,DER2)
      BX(2) = P(1)
      BX(3) = P(2)
      BX(5) = P(3)
      BX(6) = P(4)
      BY(2) = P(5)
      BY(3) = P(6)
      BY(5) = P(7)
      BY(6) = P(8)
      GO TO 33

```

C MAP3

```

31  P(1) = BX(2)
      P(2) = BX(3)
      P(3) = BY(2)
      P(4) = BY(3)
      P(5) = BX(5)
      P(6) = BX(6)
      P(7) = 0.
      CALL DIFEQ(NEL,P,DER3)
      BX(2) = P(1)
      BX(3) = P(2)
      BY(2) = P(3)
      BY(3) = P(4)
      BX(5) = P(5)
      BX(6) = P(6)
      GO TO 33

```

C ELQ

```

36  P(1) = BX(2)
      P(2) = BX(3)
      P(3) = BY(2)
      P(4) = BY(3)
      CALL DIFEQ(NEL,P,DELQ)
      GO TO 37

```

C MAP4

```

32  P(1) = BX(2)
      P(2) = BX(3)

```

```

      P(3) = BY(2)
      P(4) = BY(3)
      P(5) = 0.
      P(6) = 0.
      CALL DIFEQ(NEL,P,DER4)
37  BX(2) = P(1)
      BX(3) = P(2)
      BY(2) = P(3)
      BY(3) = P(4)
33  S = S + DAT(1)

1   CONTINUE

17  IPOS = I2
      MP2FLG = 0
      IF(NOSTOR) GO TO 5

      J = J + NCOL
      DO 14 I=1,14
14  STORE(J+I)=BETF(I)

5   CONTINUE

      IF(NOPR) RETURN

150 WRITE(3,1001) IPOS,ELNAME,S,(BX(I),I=1,3),BX(5),BX(6),
1      (BY(I),I=1,3),BY(5),BY(6)

154 CALL DASH

      RETURN

1000 FORMAT (' POS',11X,'S',7X,'QX',10X,'BX',8X,'AX',8X,'X',9X,
1  'DX',14X,'QY',10X,'BY',8X,'AY',8X,'Y',9X,'DY')

1001 FORMAT (I4,1X,A4,1X,F10.4,2(F10.6,F11.4,F11.6,F10.6,F10.6,6X))
1004 FORMAT (1H1)
1003 FORMAT (1H )
      END

C ***** TRKCN *****

      SUBROUTINE TRKCN ( MFXPT, NEL, P, NP, IOP )
      IMPLICIT REAL*8 (A-H,O-Z)

C  A ROUTINE TO CONTROL TRACKING THROUGH LINEAR-, SINGLE NON-LINEAR-,
C  OR PRD-TYPE PRODUCTS OF TRANSFORMATIONS.
C  MFXPT = INDEX OF FXPT INSTRUCTION.  HERE IT IS USED TO REFERENCE
C  THE 7X7 MATRIX REPRESENTING THE LINEARIZED BEAM SYSTEM RELATIVE TO
C  THE REFERENCE RAY PP.  ON EACH CALL TO TRKCN,  MFXPT = NEL * MFXPT.
C  NEL = INDEX OF ELEMENT OF BEAM SYSTEM
C  PP = PARTICLE VECTOR
C  NP = PARTICLE INDEX

```

```

      INCLUDE 'DIM.CCC

```

```
INCLUDE 'CONTRL.CCC'
```

```
INCLUDE 'BSTORE.CCC'
```

```
INCLUDE 'BINFF.CCC'
```

```
INCLUDE 'BWORK.CCC'
```

```
DIMENSION ISTORE(1)
```

```
EQUIVALENCE (STORE(1),ISTORE(1))
```

```
COMMON/MQ/ML
```

```
EXTERNAL DER1,DER2,DER3,DER4,DER5,DER6,DER7
```

```
DIMENSION P(7,1),T(49),PAR(25),ML(2),RW(3)
```

```
INTEGER RET,OPNAME
```

```
DATA RW/3*0./
```

```
C
```

```
NF = 0
```

```
IF (IOP.NE.1) NF = M7END - 5
```

```
NN = NEL
```

```
KIND = INFF(20,NN)
```

```
IF (KIND.EQ.4HEDRF) GO TO 35
```

```
IF (KIND.EQ.3HDKE ) GO TO 36
```

```
IF (KIND.EQ.3HSOL ) GO TO 37
```

```
IF (KIND.EQ.3HMAP) GO TO 20
```

```
IF (KIND.NE.3HPRD) GO TO 30
```

```
C PRD-TYPE PRODUCT
```

```
K = INFF(4,NN)
```

```
ASSIGN 21 TO RET
```

```
I = 1
```

```
MM = NEL
```

```
GO TO 22
```

```
C SINGLE LINEAR TRANSFORMATION
```

```
30 ASSIGN 45 TO RET
```

```
GO TO 40
```

```
C SINGLE NON-LINEAR TRANSFORMATION
```

```
20 ASSIGN 45 TO RET
```

```
IF (IOP.GT.1.AND.IOP.NE.5) ASSIGN 43 TO RET
```

```
GO TO 221
```

```
C LOOP OVER PRD FACTORS
```

```
21 I = I + 1
```

```
IF (I.GT.K) GO TO 45
```

```
22 NN = MDAT(MM,I)
```

```
KIND = INFF(20,NN)
```

```
IF (KIND.NE.3HPRD) GO TO 222
```

```
N2 = NN
```

```
K2 = 0
```

```
ASSIGN 321 TO RET
```

```
KN = INFF(4,N2)
```

```
321 K2 = K2 + 1
```

```
NN = MDAT(N2,K2)
```

```
KIND = INFF(20,NN)
```

```
322 IF (K2.EQ.KN) ASSIGN 21 TO RET
```

```
IF (KIND.NE.3HPRD) GO TO 222
```

```
N3 = NN
```



```

      K3 = 0
      ASSIGN 421 TO RET
      KN3 = INFF(4,N3)
421  K3 = K3 + 1
      NN = MDAT(N3,K3)
      KIND = INFF(20,NN)
      IF (K3.EQ.KN3) ASSIGN 321 TO RET
      IF (K2.EQ.KN.AND.K3.EQ.KN3) ASSIGN 21 TO RET
      IF (KIND.NE.3HPRD) GO TO 222
      WRITE (3,1000)
1000 FORMAT (' PRD INSTRUCTIONS NESTED MORE THAN THREE DEEP - ERROR')
      ERROR = .TRUE.
      RETURN
222  IF (KIND.NE.3HMAP) GO TO 40

C          PICK UP SUBROUTINE NAME
221  OPNAME = INFF(1,NN)
      IF (OPNAME.NE.4HSXTP) GO TO 223
      NUM = 11
      GO TO 9
223  IF (OPNAME.EQ.4HMOVE) GO TO 27
      IF (OPNAME.NE.4HNPOL) GO TO 25
      NUM=12
      GO TO 9

C          STRIP OFF MAP SUBROUTINE NUMBER
25   DECODE(4,100,OPNAME)NUM
100  FORMAT (3X,I1)
      IF (NUM.EQ.0) NUM = + 0
      NUM = NUM + 1

C          GET DATA OF MAP INSTRUCTION.  TRANSMIT IN ARGUMENT.
9    NFL = INFF(15,NN)
      CALL DATA(NN,1,1,NFL,PAR)
      GO TO (10,11,12,13,14,15,16,17,18,19,23,28), NUM
10   CALL MAP (P,PAR)
      GO TO RET,(21,43,45,321,421)

C  MAP1 — WIGGLER MAGNET
11   CALL DIFEQ(NN,P,DER1)
      GO TO RET,(21,43,45,321,421)

C  MAP2
12   CALL DIFEQ(NN,P,DER2)
      GO TO RET,(21,43,45,321,421)

C  MAP3 INTEGRATE TRANSVERSE AND LONGITUDINAL ENVELOPES.
13   CALL DIFEQ(NN,P,DER3)
      GO TO RET,(21,43,45,321,421)

C  MAP4 — ENVELOPE INTEGRATION
14   CALL DIFEQ(NN,P,DER4)
      GO TO RET,(21,43,45,321,421)

C  MAP5 — SEXTUPOLE INTEGRATION
15   CALL DIFEQ(NN,P,DER5)
      GO TO RET,(21,43,45,321,421)
16   CALL DIFEQ (NN,P,DER6)
      GO TO RET,(21,43,45,321,421)
17   CALL DIFEQ (NN,P,DER7)
      GO TO RET,(21,43,45,321,421)

```

```

18  CALL MAP8(P,PAR)
    GO TO RET,(21,43,45,321,421)
19  CALL MAP9(P,PAR)
    GO TO RET,(21,43,45,321,421)

43  CALL STOR7(MFXPT,P(1,2),RW)
    RETURN

C
23  INFF(4,NF)=INFF(4,NN)
    INFF(5,NF)=INFF(5,NN)
    CALL SXTP(NF,P,PAR)
    GO TO 26
28  CALL NPOL(NF,NN,P,PAR)
    GO TO 26

C
27  CALL MOVE(NN,NF,P)
26  IF (IOP.NE.1.AND.IOP.NE.5) GO TO 42
    GO TO RET,(21,43,45,321,421)

C  CARRY OUT A LINEAR TRANSFORMATION
40  IF (IOP.NE.1) GO TO 41
    CALL RTRV7(NN,T,RW)
    CALL MXV7(T,P,P)
    GO TO RET,(21,43,45,321,421)
41  CALL REVMAT(NF,NN,P)
    IF (IOP.EQ.5) GO TO 45
42  ML(1) = MFXPT
    ML(2) = NF
    CALL MMM ( MFXPT, 2, ML )
    IF (IOP.NE.0) CALL RTRV7(MFXPT,P(1,2),RW)
    GO TO RET,(21,43,45,321,421)

C
35  CONTINUE
    CALL EDRF(NEL,P,NP)
    GO TO 45
36  CONTINUE
    CALL DKE(NEL,P,NP)
    GO TO 45
37  CALL SOL(NEL,P,NP)

45  RETURN
    END

C***** TRKPLT *****
    SUBROUTINE TRKPLT(SCALE)
    IMPLICIT REAL*8 (A-H,O-Z)

    INCLUDE 'FILES.CCC'

C  TRK5 AND TRK6 FILES ARE DEFINE IN DATA STATEMENT IN BLOCK DATA
    DIMENSION U(7),V(7),T(3),SCALE(1)
    DATA T/1HX,1HY,1H./
    REWIND TRK5
    REWIND TRK6
    XR1 = SCALE(1)
    YU1 = SCALE(2)
    XR2 = SCALE(3)

```

```

YU2 = SCALE(4)
READ(TRK5) NPARTS,NR,NVECS,NBML,NMAT,KPOS,NVEC0,U
DO 1 K1=1,NPARTS
  NERR=0
  NDX=0
  NDY=0
  DO 2 K2=1,NR
    READ(TRK5) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
    IF(NERR.EQ.1) GO TO 2
    IF(DABS(V(1)).LT.1.D04.AND.DABS(V(2)).LT.1.D04) GO TO 4433
    K22=K2-1
    WRITE(3,777)K1,K22
777  FORMAT(1H0,'PARTICLE NUMBER',I6,' WENT UNSTABLE ON REV',I6)
    NERR=1
    GO TO 2
4433 CONTINUE
    IF(K2.GT.1) GO TO 20
    WRITE(3,100)
100  FORMAT(1H1,5X,'U1',7X,'U2',7X,'U3',7X,'U4',17X,'V1',7X,'V2',7X,
      'V3',7X,'V4',7X,'IPX  IRM  IPOS')
    WRITE(3,101) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
101  FORMAT(1H0,4X,4F9.6,6X,4F9.3,3I5)
    XL1=-XR1
    YL1=-YU1
    CALL SETP(XL1,XR1,YL1,YU1)
20   CONTINUE
    CALL PLOTT(1,T(1),V(1),V(2))
    IF(K2.EQ.NR) CALL PAGTRKP(0,K1)
    IF(DABS(V(1)).GT.XR1.OR.DABS(V(2)).GT.YU1) NDX=NDX+1
    IF(K2.EQ.NR) WRITE(3,102) NDX
102  FORMAT(1H0,I6,' PATICLES OUTSIDE DIAGRAM')
2    CONTINUE
    DO 3 K2=1,NR
      READ(TRK6) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
      IF(NERR.EQ.1) GO TO 3
      IF(DABS(V(3)).LT.1.D04.AND.DABS(V(4)).LT.1.D04) GO TO 4434
      K22=K2-1
      WRITE(3,777)K1,K22
      NERR=1
      GO TO 3
4434 CONTINUE
      IF(K2.GT.1) GO TO 30
      WRITE(3,100)
      WRITE(3,101) (U(J),J=1,4),(V(J),J=1,4),IP,IRM,IPOS
      XL2=-XR2
      YL2=-YU2
      CALL SETP(XL2,XR2,YL2,YU2)
30   CONTINUE
      CALL PLOTT(1,T(2),V(3),V(4))
      IF(K2.EQ.NR) CALL PAGTRKP(0,K1)
      IF(DABS(V(3)).GT.XR1.OR.DABS(V(4)).GT.YU2) NDY=NDY+1
      IF(K2.EQ.NR) WRITE(3,102) NDY
3    CONTINUE
1    CONTINUE
    REWIND TRK5

```

```

REWIND TRK6
RETURN
END

```

C..... SETP

```

SUBROUTINE SETP(XL,XTOP,YL,YTOP)
IMPLICIT REAL*8 (A-H,O-Z)

DIMENSION A(112,51),Z(6),X(1),Y(1)
COMMON/SAML/A,Z,XLOW,XRANGE,YLOW,YRANGE
DATA BLANK/4H /
XR=XTOP-XL
YR=YTOP-YL
XLOW=XL
XRANGE=XR
YLOW=YL
YRANGE=YR
DO 10 I=1,112
DO 10 J=1,51
A(I,J)=BLANK
10 CONTINUE
RETURN
END

```

C..... PLOTT

```

SUBROUTINE PLOTT(N,C,X,Y)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(112,51),Z(6),X(1),Y(1)
COMMON/SAML/A,Z,XLOW,XRANGE,YLOW,YRANGE
IF( N .LT. 1 )RETURN
DO 100 K=1,N
XMULT = ( X(K)-XLOW )/XRANGE
IF( XMULT .LT. 0.0 )GO TO 100
YMULT = ( Y(K)-YLOW )/YRANGE
IF( YMULT .LT. 0.0 )GO TO 100
IT = INT( 110.0*XMULT +0.500 ) +1
I=IT
IF( I .GT. 111 )GO TO 100
JT= INT( 50.00*YMULT +0.500 ) +1
J=JT
IF( J .GT. 51 )GO TO 100
A(I,J)=C
100 CONTINUE
RETURN
END

```

C..... PAGTRKP

```

SUBROUTINE PAGTRKP(IC,K1)
IMPLICIT REAL*8 (A-H,O-Z)

DIMENSION A(112,51),Z(6),X(1),Y(1)
COMMON/SAML/A,Z,XLOW,XRANGE,YLOW,YRANGE
WRITE(3,5555) K1
5555 FORMAT(1H0,'PARTICLE NUMBER',15)
DO 250 K=1,51
I=52-K
IP = MOD( I-1,5 )

```

```

      IF( IP ) 210,220,210
220  CONTINUE
      DEL=YRANGE/10.D0
      IM=I/5
      FM=IM
      XX=YLOW+FM*DEL
      IF( IC )222,221,222
221  WRITE(3,1221)XX,(A(J,I),J=1,112)
1221  FORMAT(4X,F10.3,2H +,112A1)
      GO TO 250
222  WRITE(3,1222)XX,(A(J,I),J=1,112)
1222  FORMAT(4X,E10.3,2H +,112A1)
      GO TO 250
210  WRITE(3,1210) (A(J,I),J=1,112)
1210  FORMAT(15X,1HI,112A1)
250  CONTINUE
      WRITE(3,1230)
1230  FORMAT(15X,112(1H-))
      WRITE(3,1235)
1235  FORMAT(16X,1HI,5(21X,1HI))
      DEL=XRANGE/5.D0
      DO 300 J=1,6
      Z(J)=XLOW+DEL*FLOAT(J-1)
300  CONTINUE
      IF(IC)400,500,400
500  WRITE(3,1500) (Z(I),I=1,6)
1500  FORMAT(10X,F10.3,5(12X,F10.3))
      GO TO 1499
400  WRITE(3,1400)(Z(I),I=1,6)
1400  FORMAT(10X,E10.3,5(12X,E10.3))
1499  RETURN
      END

```

C.....TRNSPS.....

```

SUBROUTINE TRNSPS
IMPLICIT REAL*8 (A-H,O-Z)

```

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
INCLUDE 'INSTR.CCC'

```

C

C MATRIX STORED BY ROWS AT LFL TRANSPOSED AND STORED BY COLUMNS AT LQ

C

```

      IL = LQ
      IRR = LFL - 1

```

C

```

      DO 1 J=1,KB
      IRR = IRR + 1
      IR = IRR
      DO 1 I=1,KA
      STORE(IL) = STORE(IR)
      IL = IL + 1
1      IR = IR + KB

```

RETURN
END

C ***** UCOPY *****

SUBROUTINE UCOPY(A,X,N)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(N),X(N)

C COMMENT COPIES N WORDS FROM A INTO X, THE BEGINNING
C OF A MAY OVERLAP THE END OF X, BUT NOT VICE-VERSA.
C MUST HAVE N.GE.0 .
IF(N.LE.0) RETURN
DO 100 I= 1,N,1
X(I)= A(I)

100 CONTINUE
RETURN
END

C ***** UPIO *****

SUBROUTINE UPIO(M)

C READS INPUT DATA, SUBSTITUTES NEW VALUES FOUND FOR
C PARA STATEMENTS AND WRITES A NEW DATA FILE .

C KA = NUMBER OF LIMIT PAIRS.
C IF KA IS BLANK OR ZERO, IT IS ASSUMED THERE ARE NO LIMITS.
C THEN ALL PARA STATEMENTS ARE UPDATED. IUPDATE IS TRUE.
C VUPDATE IS SET TRUE WHEN A BEGINNING LIMIT IS ENCOUNTERED.
C SET TO FALSE WHEN END LIMIT IS REACHED.

C WHEN STOP IS REACHED, , NO UPDATING IS ATTEMPTED. RECORDS ARE
C MERELY COPIED TO NEW FILE.

IMPLICIT REAL*8 (A-H,O-Z)
INCLUDE 'FILES.CCC'
INCLUDE 'BINFF.CCC'

DIMENSION ICARD(20)
INTEGER ELNUM,OPN
DIMENSION LIM(24),LIM1(12),LIM2(12)
LOGICAL IUPDATE,VUPDATE,SKIPUP
DATA IBLNK/1H /
REWIND IFILE

IUPDATE = .TRUE.
SKIPUP = .FALSE.

KA = INFF(4,M)
IF (KA.EQ.0) GO TO 101
IUPDATE = .FALSE.
NUM = INFF(17,M)
CALL DATA (M,2,1,NUM,LIM)
II = 1
DO 10 I=1,NUM,2
LIM1(II) = LIM(I)
LIM2(II) = LIM(I+1)
10 II = I + 1
VUPDATE = .FALSE.

```

101 READ (IFILE,1000,END=700) (ICARD(I),I=1,20)
    DECODE (4,3,ICARD) ISIGN
3   FORMAT (A1,3X)
    IF (ISIGN.EQ.1HP.OR.ISIGN.EQ.1HC.OR.ISIGN.EQ.1H.) GO TO 104
    DECODE (20,2,ICARD) ISGN,NM,OPN,KAM,KBM
2   FORMAT (A1,A4,2X,A4,1X,A3,1X,A3,1X)
    IF (SKIPUP) GO TO 104
    IF (IUPDATE) GO TO 100
    IF (VUPDATE) GO TO 30
C   VUPDATE IS FALSE. TEST FOR A TURN ON LIMIT
    DO 20 I=1,KA
        IF (NM.EQ.LIM1(I)) GO TO 25
20  CONTINUE
    GO TO 102
25  VUPDATE = .TRUE.
    IX = I
C   VUPDATE IS TRUE. TEST FOR TURN OFF LIMIT
30  IF (NM.NE.LIM2(IX)) GO TO 100
    VUPDATE = .FALSE.

100 IF (OPN.EQ.4HPARA) GO TO 500
102 WRITE (OFILE,1001) ISGN,NM,OPN,KAM,KBM,
    1 (ICARD(IW),IW=6,20)
    IF (OPN.EQ.4HSTOP) SKIPUP = .TRUE.
    GO TO 101
104 WRITE (OFILE,1000) (ICARD(I),I=1,20)
    GO TO 101

C   PARA STATEMENT.
500 CONTINUE
    MM = ELNUM(NM)
    CALL DATA (MM,1,1,1,FVAR)
    FV = DABS(FVAR)
C   SCALE VALUE TO CARRY MAXIMUM FIGURES.
    IF (FVAR.EQ.0) GO TO 625
    IF (FV.LT.1.) GO TO 575

C   VALUES GREATER THAN 1.
    DO 505 I = 1,200
        KV = I
        FAC = 10.**I
        IF (FV.LT.FAC) THEN
            FVAR = FVAR/FAC
            GO TO 650
        ENDIF
505 CONTINUE
C   VALUE OUT OF BOUNDS. PRINT ERROR MESSAGE
    WRITE (3,510) FVAR
510 FORMAT (5X,'*****',2X,F15.5,2X,'OUT OF BOUNDS. ')
    WRITE (OFILE,510) FVAR
    GO TO 625

C   VALUE IS LESS THAN 1.
575 DO 580 I = 1,200
    KF = -I

```

```

FC = 10.**KF
IF (FV.GE.FC) THEN
  KV = KF + 1
  KF = -KV
  FAC = 10.**KF
  FVAR = FVAR*FAC
  IF (KV.EQ.0) GO TO 625
  GO TO 650
ENDIF
580 CONTINUE
C VALUE OUT OF BOUNDS. PRINT ERROR MESSAGE.
C SHOULD HAVE HUNG UP BEFORE THIS.
  WRITE (3,581) FVAR
  WRITE (OFILE,581) FVAR
  FVAR = 0.
581 FORMAT (5X,'*****',1X,F15.12,'OUT OF BOUNDS.')
625 WRITE (OFILE,1003) ISGN,NM,OPN,KAM,KBM,FVAR
  GO TO 101
650 WRITE (OFILE,1002) ISGN,NM,OPN,KAM,KBM,FVAR,KV
  GO TO 101

700 RETURN

1000 FORMAT (20A4)
1001 FORMAT (A1,A4,2X,A4,1X,A3,1X,A3,1X,15A4)
1002 FORMAT (A1,A4,2X,A4,1X,A3,1X,A3,1X,F10.7,15)
1003 FORMAT (A1,A4,2X,A4,1X,A3,1X,A3,1X,F10.7)
  END

C ***** VERMIN *****
  SUBROUTINE VERMIN(A,L,M,N,IFAIL)
  IMPLICIT REAL*8 (A-H,O-Z)
CC      INVERTS A SYMMETRIC MATRIX. MATRIX IS FIRST SCALED TO
CC      HAVE ALL ONES ON THE DIAGONAL (EQUIVALENT TO CHANGE OF UNITS)
CC      BUT NO PIVOTING IS DONE SINCE MATRIX IS POSITIVE-DEFINITE.
CC
  DIMENSION A(L,M),PP(15),Q(15),S(15)
  COMMON /PARINT/ X(15),XT(15),DIRIN(15),MAXINT,NPAR
  IFAIL=0
  IF (N.LT. 1) GO TO 100
  IF (N.GT. MAXINT) GO TO 100
C      SCALE MATRIX BY SQRT OF DIAG ELEMENTS
  DO 8 I=1,N
    SI = A(I,I)
    IF (SI) 100,100,8
  8 S(I) = 1.0D0/DSQRT(SI)
  DO 20 I= 1, N
    DO 20 J= 1, N
  20 A(I,J) = A(I,J) *S(I)*S(J)
C      . . . START MAIN LOOP . . .
  DO 65 I=1,N
    K = I
C      PREPARATION FOR ELIMINATION STEP1
    Q(K)=1.0D0/A(K,K)
    PP(K) = 1.0D0

```



```

      A(K,K)=0.0D0
      KP1=K+1
      KM1=K-1
      IF(KM1)100,50,40
40 DO 49 J=1,KM1
      PP(J)=A(J,K)
      Q(J)=A(J,K)*Q(K)
49 A(J,K)=0.0D0
50 IF(K-N)51,60,100
51 DO 59 J=KP1,N
      PP(J)=A(K,J)
      Q(J)=-A(K,J)*Q(K)
59 A(K,J)=0.0D0
C          ELIMINATION PROPER
60 DO 65 J=1,N
      DO 65 K=J,N
65 A(J,K)=A(J,K)+PP(J)*Q(K)
C          ELEMENTS OF LEFT DIAGONAL AND UNSCALING
      DO 70 J= 1, N
      DO 70 K= 1, J
      A(K,J) = A(K,J) *S(K)*S(J)
70 A(J,K) = A(K,J)
      RETURN
C          FAILURE RETURN
100 IFAIL=1
      RETURN
      END
C ***** WBE *****
      SUBROUTINE WBE(K,L)
      IMPLICIT REAL*8 (A-H,O-Z)
C  WRITES BETATRON FUNCTIONS OF ELEMENTS MI(I),I=1,K
      INCLUDE 'BINFF.CCC'
      INCLUDE 'BMI.CCC'
      INCLUDE 'CONST.CCC'

      DIMENSION BETX(9),BETY(9)
      CALL HED
      IF (L.NE.0) GO TO 40
      WRITE(3,51)
      GO TO 42
40  WRITE(3,52)
42  DO 45 I=1,K
      M1=MI( I )
      N=INFF(2,M1)
      IF (L.NE.0) GO TO 50
      CALL BET(M1,BETX,BETY,1)
      BETX(1) = BETX(1) / PI2
      BETY(1) = BETY(1) / PI2
      WRITE(3,53) N,BETX(1),BETX(2),BETX(3),BETX(5),BETX(6),BETX(7),
1BETY(1),BETY(2),BETY(3),BETY(5),BETY(6),BETY(7)
      GO TO 45
50  CALL BET(M1,BETX,BETY,0)
      BETX(1) = BETX(1) / PI2
      BETY(1) = BETY(1) / PI2
      WRITE(3,53) N,BETX(1),BETX(2),BETX(3),BETX(5),BETX(6),BETX(9),

```

```

1BETY(1),BETY(2),BETY(3),BETY(5),BETY(6),BETY(9)
45  CONTINUE
    CALL HED
    RETURN
51  FORMAT( 5X,19HBETATRON FUNCTIONS./          1H0,131HNAME      MU
1X/2PI  BETAX  ALPHAX      XEQ      DXEQ      WX      MUY/
22PI  BETAY  ALPHAY      YEQ      DYEQ      WY      /1H0)
52  FORMAT( 5X,19HBETATRON FUNCTIONS./1H0,131HNAME      MUX/2PI  BETAX
1  ALPHAX      XEQ      DXEQ      XTRACE/2      MUY/2PI  BETAY
2  ALPHAY      YEQ      DYEQ      YTRACE/2 /1H0)
53  FORMAT(1X,A5,F12.8,F10.5,4F10.6,2X,F12.8,F10.5,4F10.6)
    END
C  ..... WMA .....
    SUBROUTINE WMA(KA)
    IMPLICIT REAL*8 (A-H,O-Z)
C    WRITES MATRICES OF ELEMENTS MI(1),I=1,KA.
    INCLUDE 'BINFF.CCC'
    INCLUDE 'SWTCH.CCC'
    INCLUDE 'BMI.CCC'

    INTEGER doIR7/2HR7/,doIROT/3HROT/
    DIMENSION RX(2,3),RY(2,3),RW(3),T(7,7)
C    PRINT ALL MATRICES AS DEFINED BY SIZE IF SIZE ARBITRARY.
C    PRINT IN 7X7 IF ANY ARE 7X7

    CALL DASH
    IF (MSIZE.EQ.7) GO TO 300
    IF (MSIZE.EQ.3) GO TO 102
    DO 100 I=1,KA
    M1 = IABS(MI(I))
    KN = INFF(20,M1)
    IF (KN.EQ.doIR7.OR.KN.EQ.doIROT) GO TO 300
100  CONTINUE
C  PRINT ALL AS 3X3 MATRICES
102  IF (BEND.NE.0.) GO TO 105
C  PRINT HEADING FOR MISALIGNMENT
    WRITE (3,11)
    GO TO 106
C  PRINT HEADING FOR TRANSFER MATRICES—NO MISALIGNMENT
105  WRITE (3,21)
106  DO 150 I=1,KA
    M1 = MI(I)
    N = MNAME(M1)
    CALL RXY(M1,RX,RY,RW)
    IF (BEND.NE.0.) GO TO 110
    WRITE (3,12) (RX(1,J),J=1,3),(RY(1,J),J=1,3)
    WRITE (3,13) N, (RX(2,J),J=1,3), (RY(2,J),J=1,3)
    GO TO 120
110  WRITE (3,22) (RX(1,J),J=1,3),(RY(1,J),J=1,2)
    WRITE (3,23) N, (RX(2,J),J=1,3), (RY(2,J),J=1,2)
    W1 = RX(1,3)*RX(2,1) - RX(2,3)*RX(1,1)
    W2 = RX(1,3)*RX(2,2) - RX(2,3)*RX(1,2)
    WRITE (3,24) W1,W2,RW(3)
120  WRITE (3,15) RW(1),RW(2)
150  CONTINUE

```

```

      GO TO 355
C PRINT ALL AS 7X7 MATRICES
300 CONTINUE
      WRITE (3,36)
      DO 350 I=1,KA
      M1 = MI(I)
      N = MNAME(M1)
      CALL RTRV7(M1,T,RW)
      DO 305 II=1,3
      WRITE (3,34) (T(II,J),J=1,7)
305 CONTINUE
      WRITE (3,35) N,(T(4,J),J=1,7)
      DO 310 II=5,7
      WRITE (3,34) (T(II,J),J=1,7)
310 CONTINUE
      WRITE (3,15) RW(1),RW(2)
350 CONTINUE
355 CALL DASH
      RETURN
11  FORMAT (20H  TRANSFER MATRICES,25X,7HRX(I,J),40X,7HRY(I,J)//
134X,1HX,11X,2HDX,11X,1H1,20X,1HY,12X,2HDY,11X,1H1/9X,7HELEMENT)
12  FORMAT (27X,3F13.8,8X,3F13.8)
13  FORMAT (10X,A5,12X,3F13.8,8X,3F13.8/30X,2H0.,11X,2H0.,11X,2H1.,19X
1,2H0.,11X,2H0.,11X,2H1./)
15  FORMAT (/30X,9HLENGTH = ,F13.8,5X,9H THETA = ,F13.8///)
21  FORMAT (20H  TRANSFER MATRICES,29X,7HRX(I,J),35X,7HRY(I,J)//
134X,1HX,9X,5HDX/DS,9X,3H-DS,8X,4HDP/P,15X,1HY,10X,5HDY/DS/
29X,7HELEMENT)
22  FORMAT (27X,2F13.8,3X,2H0.,8X,F13.8,3X,2F13.8)
23  FORMAT (10X,A5,12X,2F13.8,3X,2H0.,8X,F13.8,3X,2F13.8)
24  FORMAT (27X,2F13.8,3X,10H1.00000000,F13.8/30X,2H0.,11X,2H0.,11X,
12H0.,11X,10H1.00000000)
34  FORMAT (30X,7F13.8)
35  FORMAT (10X,A5,15X,7F13.8)
36  FORMAT (20H  TRANSFER MATRICES,53X,6HR(I,J)//36X,1HX,10X,5HDX/DS,
110X,1HY,10X,5HDY/DS,9X,3H-DS,9X,4HDP/P,10X,1H1/9X,7HELEMENT)
      END
C ***** WRTINF *****
      SUBROUTINE WRTINF(INF,M)
      IMPLICIT REAL*8 (A-H,O-Z)
C WRITES ONE COLUMN OF INF INTO INFF(M)
      INCLUDE 'BINFF.CCC'
      DIMENSION INF(1)
      INTEGER doIASTRK/4H* /
      DO 20 I=1,24
      IF(INF(I).EQ.doIASTRK) GO TO 20
      INFF(I,M)=INF(I)
20 CONTINUE
      RETURN
      END
C ***** WFLSR *****
      SUBROUTINE WFLSR(M)
      IMPLICIT REAL*8 (A-H,O-Z)
C KA = 0 EXECUTES WFL ON ERROR ONLY.
C KB = 0 WRITES CONTENTS OF INFF ONLY.

```

```

C  KB = 1 WRITES CONTENTS OF STORE, BUT NOT WORKING STORAGE.
C  KB = 2 ALSO WRITES WORKING STORAGE.
C  WRITE CONTENTS OF INFF(J,1),J=1,24 AND I = MIN,MAX
C  WRITES CONTENTS OF STORE USED BY INSTRUCTIONS.
C  WORKING STORAGE PRINTED ALSO IF KB.GT.1.
    INCLUDE 'BSTORE.CCC'
    INCLUDE 'BINFF.CCC'
    INCLUDE 'BWORK.CCC'
    DIMENSION ISTORE(1)
    EQUIVALENCE (STORE(1),ISTORE(1))
    INCLUDE 'CONTRL.CCC'
    INCLUDE 'STORE.CCC'
    INCLUDE 'DIM.CCC'
    INTEGER BDAT,ELNUM

    LOGICAL*1 ISGN,BLNK/' '/,DASH/'-'/
    LOGICAL ENDL
    CHARACTER*8 BLANK,SBLANK

C
    DATA BLANK/'      '/
    DATA IBLNK/'    '/
    EQUIVALENCE (BBLANK,SBLANK)

    IOP = INFF(3,M)
    IF (IOP.LT.0) RETURN
    KB = INFF(5,M)
    NBC = INFF(17,M)
    LBC = INFF(16,M)
    BBLANK = STORE(LBC)
    IF (SBLANK.NE.BLANK) GO TO 50

C  PRINT INFF
16  WRITE (3,24)
24  FORMAT (/2X,25H CONTENTS OF      INFF(I,M))
    NSAV = ISAV + ISAV7
    MX = MAX
    IF (KB.LT.2) MX = MX - NSAV
    NUM=MX-MIN+1
    DO 10 I=1,NUM
    WRITE (3,22) INFF(2,MX),INFF(1,MX),MX,(INFF(J,MX),J=1,4)
22  FORMAT (/2X,2(A5,2X),9H((M(HEX)=,Z5,8H),I=1,4),4(2X,Z20))
    WRITE (3,23) MX,(INFF(J,MX),J=5,8)
23  FORMAT (16X,9H(M((DEC)=,I5,8H),I=5,8),4(2X,Z20))
    J1 = 9
    DO 31 K=1,4
    J2 = J1+ 3
    WRITE(3,30) J1,J2,(INFF(J,MX),J=J1,J2)
    J1 = J2 + 1
30  FORMAT (21X,3H(I=,I2.1H ,I2,1H),5X,4(2X,Z20))
31  CONTINUE
    MX=MX-1
10  CONTINUE
C  PRINT STORE ON OPTION
    IF (KB.EQ.0) RETURN
    LX=LFILE

```

```

      L = ISAV*(12+KADDR)+ ISAV7*(49+KADD)
      IF (KB.GT.1) L=1
C     ENDL=.FALSE.
      WRITE (3,28)
28    FORMAT (37H1   CONTENTS OF STORE IN HEXIDECIMAL//
153H  HEX LOCATION      PLUS  0  1  2  3  4  5  6  7/
253H                                8  9  A  B  C  D  E  F//)
17    J=L+3
      IF (J.LT.LX) GO TO 18
      J=LX
      ENDL=.TRUE.
18    WRITE (3,25) L,(STORE(I),I=L,J)
25    FORMAT (5X,Z8,5X,4(Z20,2X))
      IF (ENDL) GO TO 20
      L=J+1
      J=L+3
      IF (J.LT.LX) GO TO 19
      J=LX
      ENDL=.TRUE.
19    WRITE (3,26) (STORE(I),I=L,J)
26    FORMAT (16X,4(Z20,2X)//)
      IF (ENDL) GO TO 20
      L=J+1
      GO TO 17
20    WRITE (3,27)
27    FORMAT (1H1)
      RETURN

C  PRINT INFF AND STORE OF SELECTED ELEMENTS
50    CONTINUE
      DO 80 I=1,NBC
C    PRINT INFF
      NAM = BDAT(M,I)
      IF (NAM.EQ.IBLNK) GO TO 80
      WRITE (3,24)
      MX = ELNUM(NAM)
      WRITE (3,22) INFF(2,MX),INFF(1,MX),MX,(INFF(J,MX),J=1,4)
      WRITE (3,23) MX,(INFF(J,MX),J=5,8)
      J1 = 9
      DO 51 K=1,4
      J2 = J1+ 3
      WRITE(3,30) J1,J2,(INFF(J,MX),J=J1,J2)
      J1 = J2 + 1
51    CONTINUE
C  FIND STARTING POINT OF STORAGE FOR THIS ELEMENT
      MTOT = INFF(18,MX)
      IF (MTOT.EQ.0) GO TO 80
      MLQ = INFF(10,MX)
      IF (MLQ.NE.0) GO TO 60
      MLQ = INFF(14,MX)
      IF (MLQ.NE.0) GO TO 60
      MLQ = INFF(16,MX)
      IF (MLQ.NE.0) GO TO 60
      MLQ = INFF(12,MX)
60    CONTINUE

```

```

        WRITE (3,61)
61      FORMAT (/6X,'CONTENTS OF STORE IN HEXIDECIMAL'/)
        LL = MLQ
        J = 0
65      JX = LL + 3
        WRITE (3,25) LL,(STORE(K),K=LL,JX)
        J = J + 4
        IF (J.GE.MTOT) GO TO 80
        LL = LL + 4
        GO TO 65
80      CONTINUE
        WRITE (3,100)
100     FORMAT (/)
        RETURN
        END

```

C XEQCON

```

        SUBROUTINE XEQCON(M)
        IMPLICIT REAL*8 (A-H,O-Z)
C      CONTROL EXECUTION OF SYNCH INSTRUCTION BE ACT OR DELE
C      IF KB=1, KA ELEMENTS SHOULD BE CHANGED BEGINNING WITH ONE INPUT
C      IF KB = 0, INPUT CONTAINS KA ELEMENTS
        INCLUDE 'BSTORE.CCC'
        INCLUDE 'BINFF.CCC'
        INCLUDE 'BWORK.CCC'
        DIMENSION ISTORE(1)
        EQUIVALENCE (STORE(1),ISTORE(1))
        LOGICAL SEQ,ACT
        INTEGER ELNUM
        DIMENSION NDAT(50)
        INTEGER do1DELE/'DELE'/

        SEQ = .FALSE.
        KA = INFF(4,M)
        KB = INFF(5,M)
        NAM = INFF(1,M)
        ACT = .TRUE.
        IF (NAM.EQ.do1DELE) ACT = .FALSE.
        IF (KB.EQ.1) SEQ = .TRUE.
        KC = KA
        IF (SEQ) KC = 1
        CALL DATA(M,2,1,KC,NDAT)
        IF(SEQ) GO TO 15
C      CHANGE EXECUTION MODE OF ALL ELEMENTS IN NDAT ARRAY
        DO 10 I=1,KA
        NN = ELNUM(NDAT(I))
        IOP = IABS(INFF(3,NN))
        IF (.NOT.ACT) IOP = -IOP
        INFF(3,NN) = IOP
10      CONTINUE
        RETURN
15      CONTINUE
        NN = ELNUM(NDAT(1))
        DO 20 I=1,KA
        IOP = IABS(INFF(3,NN))

```

```

      IF (.NOT.ACT) IOP = -IOP
      INFF(3,NN) = IOP
      NN = INFF(8,NN)
20    CONTINUE
      RETURN
      END

C ..... ZAM .....
      SUBROUTINE ZAM(X,DERI,Y,F,T,SWPR)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION F(1),T(8,1)
      DIMENSION Y(1)
      COMMON /INTC/ AREF,EMAX,SSSR,HFAC,SWAM,SWEX,IPMX
      COMMON /INTP/ HPR,XX,EUB,ELB,SWIN,NEQS,IP,IT,NRKS
      LOGICAL SWAM,SWEX,SWIN
      LOGICAL SWPR
      INTEGER HFAC
      REAL*16 T,HPR,XX,D,H
6000  FORMAT (36H0 CANNOT DECREASE H BECAUSE OF HMIN. ,1PE16.8,120)
C
1    CONTINUE
      SWPR=.FALSE.
      TEST=0.0
      H=HPR/QFLOAT(IP*24)
      IF ((NRKS .LT. 3) .OR. (.NOT. SWAM)) GO TO 200
C
C    ADAMS-MOULTON STEP.
100  CONTINUE
      DO 109 I=1,N
      D=QEXTD(F(I))
      T(4,I)=D
      Y(I)=DBLEQ(T(5,I)+H*(
X 55.0Q0*D-59.0Q0*T(3,I)+37.0Q0*T(2,I)- 9.0Q0*T(1,I) ))
109  CONTINUE
      X=DBLEQ(XX+24.0Q0*H)
      CALL DERI (X,Y,F)
      DO 119 I=1,N
      D=QEXTD(F(I))
      D=(      T(5,I)+H*(
X 9.0Q0*D+19.0Q0*T(4,I)- 5.0Q0*T(3,I)+      T(2,I) ))
      T(6,I)=D
      E=DABS(DBLEQ(D)-Y(I))/14.D0
      TEST=DMAX1(E/DMAX1(AREF,DABS(DBLEQ(D))),TEST)
119  CONTINUE
C
      GO TO 300
C
C    ZONNEVELD STEP.
200  CONTINUE
      DO 209 I=1,N
      D=QEXTD(F(I))
      T(4,I)=D
C
1    Y(I)=DBLEQ(T(5,I)+H*(
X 12.0Q0*D
      ))
209  CONTINUE

```

```

      X=DBLEQ(XX+12.0Q0*H)
      CALL DERI (X,Y,F)
      DO 219 I=1,N
      D=QEXTD(F(I))
      T(6,I)=D
C      2
      Y(I)=DBLEQ(T(5,I)+H*(
X 12.0Q0*D                                ))
219  CONTINUE
      CALL DERI (X,Y,F)
      DO 229 I=1,N
      D=QEXTD(F(I))
      T(7,I)=D
C      3
      Y(I)=DBLEQ(T(5,I)+H*(
X 24.0Q0*D                                ))
229  CONTINUE
      X=DBLEQ(XX+24.0Q0*H)
      CALL DERI (X,Y,F)
      DO 239 I=1,N
      D=QEXTD(F(I))
      T(8,I)=D
C      4
      Y(I)=DBLEQ(T(5,I)+H*(
X 3.75Q0*T(4,I)+5.25Q0*T(6,I)+9.75Q0*T(7,I)-0.75Q0*D ))
239  CONTINUE
      X=DBLEQ(XX+18.0Q0*H)
      CALL DERI (X,Y,F)
      DO 249 I=1,N
      D=QEXTD(F(I))
      E=DABS(DBLEQ(H*(
X -16.0Q0*T(4,I)+48.0Q0*T(6,I)+48.0Q0*T(7,I)+48.0Q0*T(8,I)
X -128.0Q0*D      )))
C      5
      D=(      T(5,I)+H*(
X 4.0Q0*T(4,I)+ 8.0Q0*T(6,I)+ 8.0Q0*T(7,I)+ 4.0Q0*T(8,I)
X ))
      T(6,I)=D
      TEST=DMAX1(E/DMAX1(AREF,DABS(DBLEQ(D))),TEST)
249  CONTINUE
C
C      BOTH ADAMS-MOULTON AND ZONNEVELD METHODS CONTINUE FROM HERE.
300  CONTINUE
      X=DBLEQ(XX+24.0Q0*H)
      IF (TEST .LE. EUB) GO TO 310
      IF (IP*HFAC .GT. IPMX) GO TO 309
C
C      REPEAT STEP WITH SMALLER H.
      NRKS=0
      IP=IP*HFAC
      IT=IT*HFAC
      DO 305 I=1,N
      Y(I)=DBLEQ(T(5,I))
      F(I)=DBLEQ(T(4,I))
305  CONTINUE

```



```

      GO TO 1
C
C      CANNOT DECREASE H BECAUSE OF HMIN.
309  CONTINUE
      IF (.NOT. SWIN) GO TO 310
      WRITE(3,6000) X,IPMX
      SWIN=.FALSE.
C
310  CONTINUE
C
C
C      ACCEPT CURRENT STEP.
C
C      XX STILL HAS NOT BEEN CHANGED SINCE ENTRY.
C      YY(XX) IS STILL IN T(5, ).
C      F(YY) IS IN T(4, ).
C
      IT=IT+1
      XX=XX+HPR/QFLOAT(IP*24)
      NRKS=MIN0(NRKS+1,4)
      DO 319 I=1,N
        D=T(6,I)
        T(5,I)=D
        Y(I)=DBLEQ(D)
319  CONTINUE
      X=DBLEQ(XX)
      CALL DERI (X,Y,F)
      IF (IT .LT. IP) GO TO 320
C
C      X IS A MULTIPLE OF HPRINT.
      SWPR=.TRUE.
      IT=IT-IP
C
320  CONTINUE
      IF (TEST .GE. ELB) GO TO 330
      IF (MOD(IP,HFAC)+MOD(IT,HFAC) .NE. 0) GO TO 330
C
C      PROCEED TO NEXT STEP WITH LARGER H, USING ZONNEVELD METHOD.
      NRKS=0
      IP=IP/HFAC
      IT=IT/HFAC
      RETURN
C
C
C      PROCEED TO NEXT STEP WITH SAME H.
330  CONTINUE
      DO 339 I=1,N
        T(1,I)=T(2,I)
        T(2,I)=T(3,I)
        T(3,I)=T(4,I)
339  CONTINUE
      RETURN
      END
C ***** ZAM0 *****
      SUBROUTINE ZAM0 (NO,X,DERI,Y,F,T,HPR0)

```

```

IMPLICIT REAL*8(A-H,O-Z)
COMMON /INTC/ AREF,EMAX,SSSR,HFAC,SWAM,SWEX,IPMX
COMMON /INTP/ HPR,XX,EUB,ELB,SWIN,NEQS,IP,IT,NRKS
LOGICAL SWAM,SWEX,SWIN
DIMENSION Y(1),F(1),T(8,1)
INTEGER HFAC
REAL*16 T,HPRO,HPR,XX

```

C

```

HPR=HPRO
XX=QEXTD(X)
N=NO
EUB=EMAX
ELB=EMAX/SSSR
IP=1
IT=0
NRKS=0
SWIN=SWEX
CALL DERI (X,Y,F)
DO 9 I=1,N
  T(5,I)=QEXTD(Y(I))
9 CONTINUE
RETURN
END

```

File: SYNPL.FOR;1

```

SUBROUTINE PLTBET(M)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

C*****
C NAM  BEP  KA  KB  MODE IP1  IP2  BMAX  QMIN  QMAX
C*****
C PLOT BETAX,BETAY,XEQ, WHICH WERE COMPUTED AND WRITTEN TO A FILE BY
C A BEST INSTRUCTION.

```

```

INCLUDE 'BSTORE.CCC'
INCLUDE 'BINFF.CCC'
INCLUDE 'BWORK.CCC'
DIMENSION ISTORE(1)
EQUIVALENCE (STORE(1),ISTORE(1))
INCLUDE 'XXYY.CCC'
INCLUDE 'BPLTCOM.CCC'
INCLUDE 'BMI.CCC'
INCLUDE 'TRKINT.CCC'
DIMENSION IPRS(2),PMX(3),FLT(3),PAR(3),ITIK(3)
DIMENSION IFLT(3),MOP(8)
REAL*4  XPL(2000),YPL(2000),X(5),Y(5),XMIN,XMAX,YMIN,YMAX
REAL*4  PATHX,PATHY,PX,PY,FTK

```

```

DATA MOP/3HDRF,3HMAG,4HMAGV,4HMAGS,3HREF,3HINV,3HSHF,3HEQU/
CHARACTER*(*)  doIJOIN,doIBETAX,doIBETAY,doIW,doIX,doIY,

```

```

1          doIPLNTH
PARAMETER   ( doIJOIN      = 'JOIN$'      ,
1          doIBETAX       = 'BETAX$'      ,
1          doIBETAY       = 'BETAY$'      ,
1          doIW           = 'W$'          ,

```

```

1          do!X          = 'X$'          ,
1          do!Y          = 'Y$'          ,
1          do!PLNTH      = 'PATH LENGTH (M)$' )

      INTEGER BDAT
      LOGICAL FOCUS,LIMSW
C*** INPUT PARAMETERS
C      KA IS NUMBER OF FRAMES TO BE PLOTTED. THE LIMITS OF EACH FRAME
C      DENOTED BY IP1 AND IP2, WHICH ARE POSITIONS IN BETATRON TABLE.
C      KB IS MULTIPLE OF 10 INCHES FOR LENGTH OF FRAME.
C      NC IS NAME OF ELEMENT THAT COMPUTED THE BETATRON FUNCTIONS.
C      IF MODE IS TERM, THEN RUN GRAPH INTERACTIVELY AND HAVE IT APPEAR ON
C      TERMINAL. IF MODE IS ZETA, THEN PREPARE GRAPH FILE FOR ZETA. (IT MUST
C      ACTUALLY BE SENT, USING PLOTSEND, BY THE USER TO THE ZETA.)
C      IF MODE FIELD IS BLANK, THE DEFAULT IS INTERACTIVE GRAPH.
C****

C** COMPUTED VARIABLES
C      XX = LENGTH S      Y1=BETAX, Y2=XEQ, Y3=BETAY, Y4 = YEQ
C      PMX = MAX OF BETA, MAX OF XEQ,YEQ, AND MIN OF XEQ,YEQ
C      FLT = RADJUSTED MAX OF BETAX,BETAY, THE READJUSTED MAX OF XEQ, AND
C      THE READJUSTED MIN OF XEQ
C***

      LIMSW = .FALSE.
      BXXMX = BXX
      BYMX = BYX

C      GET INPUT DATA
      KA = INFF(4,M)
      KB = INFF(5,M)
      IF (KA.EQ.0) KA = 1
      NC = BDAT(M,1)
      MODE = BDAT(M,2)
      NAM = MDAT(M,1)
      KO = 2
      CALL DATA (M,3,1,2,IPRS)
      CALL DATA(M,1,1,3,PAR)
C      SET UP MI LIST OF BEAM
      NDAT=INFF(17,NAM)
      CALL MIFILL(NAM,1,NDAT,NELS,MI)
      NK=NELS
      IF (MN.EQ.NC) GO TO 36
      write(3,35)
35 format(/' bep and best donot match'//')
      stop 77
36      NPT = KW
      NZ = KW - 1

      IF (KB.NE.0) GO TO 33
C      TAKE SQUARE ROOTS OF BETAX,BETAY
      BXXMX = DSQRT(BXXMX)
      BYMX = DSQRT(BYMX)
      DO 32 I=1,KW
      Y1(I) = DSQRT(Y1(I))
      Y3(I) = DSQRT(Y3(I))

```

```

32  CONTINUE
33  CONTINUE

C      DETERMINE PAPER COORDINATES
C      LENGTH OF PAPER MAY NOT EXCEED 120 INCHES
      IF (KA.GT.14) KA=14

C      ALLOW SPACE FOR RIGHT AND LEFT MARGINS.
      FKA = KA

C.....
C      FIND MAXIMUMS AND MINIMUMS. PUT BETAX AND BETAY ON SAME SCALE.
C      MINIMUM OF BETAS IS 0. XEQ,YEQ MAY BE NEGATIVE

C      COMPUTE MAXIMUM LIMITS OF BETAS AND EQS. THEN CHECK INPUT.
C      IF ANY ARE INPUT, ALL SHOULD BE THERE
      PMX(1) = BXXM
      IF (PYMX.GT.PMX(1)) PMX(1) = BYMX
      QMX = Y2(1)
      QMN = Y2(1)
      DO 40 I=1,NPT
      IF (Y4(I).GT.QMX) QMX = Y4(I)
      IF (Y2(I).GT.QMX) QMX = Y2(I)
      IF (Y4(I).LT.QMN) QMN = Y4(I)
      IF (Y2(I).LT.QMN) QMN = Y2(I)
40  CONTINUE
      PMX(2) = QMX
      PMX(3) = QMN
C      CHECK ON INPUT MAXIMUMS
      IF (PAR(1).NE.0.) GO TO 41
      IF (PAR(3).EQ.0.) GO TO 42
C      USE INPUT LIMITS, UNLESS MAX IS 0.
41  IF (PAR(1).NE.0.) PMX(1) = PAR(1)
      IF (PAR(3).NE.0.) PMX(2) = PAR(3)
      PMX(3) = PAR(2)
      LIMSW = .TRUE.
42  CONTINUE
C      ROUND LIMITS AND SET TICK INTERVALS
      DO 55 I=1,2
      CALL PLPAR(PMX(I),FM,ITK)
      FLT(I) = FM
      ITIK(I) = ITK
55  CONTINUE
C      ROUND FOR NEGATIVE SCALE OF XEQ.
C      IF LESS THAN 10, ROUND TO NEXT INTEGER
C      IF GREATER THAN 10, ROUND TO NEXT FACTOR OF 10.
      IF (PMX(3).GE.0.) GO TO 63
      ABP = DABS(PMX(3))
      IF (LIMSW) GO TO 57
C      IF NEGATIVE QUANTITY IS LESS THAN 1 PERCENT OF MAXIMUM, SET
C      MINIMUM TO ZERO.
      ONEP = .01*QMX
      IF (ABP.LE.ONEP) GO TO 63
57  IF (ABP.GT.10.) GO TO 62
      IP = ABP

```

```

APP = IP
IF (ABP.GT.APP) IP = IP + 1
FLT(3) = -IP
GO TO 65
62 IP = ABP/10.
ABP = (IP+1)*10
FLT(3) = -ABP
GO TO 65
63 FLT(3) = 0.
PMX(3) = 0.
65 CONTINUE
DO 66 I=1,3
IFLT(I) = FLT(I)
66 CONTINUE

C          INITIAL PLOTTING DEVICE, PARAMETERS AND
C          DETERMINE LIMITS IN X SCALE
IF (IPRS(1).GT.IPRS(2)) GO TO 205
call pitinit(mode,5,6,19)
IF (IPRS(2).NE.0.) GO TO 70
C* DEFAULT — ENTIRE BEAM LINE
IPRS(1) = 0
IPRS(2) = NZ

C*****
C  ORIGINALLY SET UP TO LOOP FROM HERE TO 200 TO PLOT MULTIPLE FRAMES
C  INPUT CHANGED TO REQUIRE CARD INPUT FOR EACH FRAME.
C*****
70 DO 200 KK=1,KO,2
N1 = IPRS(KK) + 1
NBT = IPRS(KK+1) + 1
NBE = NBT - 1
NF = NBT - N1 + 1
C          LIMITS OF X FROM S1 TO SND
S1 = XX(N1)
SND = XX(NBT)
J1 = N1

C          SET UP PHYSICAL PLOT AREA
PAGEX = FKA*8.5
PAGEY = 11.
XAXIS = PAGEX - 1.6
YAXIS = PAGEY - 1.6
CALL PAGE( PAGEX, PAGEY )
CALL AREA2D( XAXIS, YAXIS )
call height(.3)

C          DRAW BOUNDARIES

YMAX = PMX(1)
YMIN = 0.
XMAX = SND
XMIN = S1
CALL XNAME( doiPLNTH, 100 )
if(kb.ne.0) then

```

```

      call yname('BETA$',100)
    else
      call yname('SQRT(BETA)$',100)
    endif

```

C*****

C SET TICKS FOR BETAX AND BETAY ON LEFT Y AXIS

```

      FTK = FLT(1)
      ITK = ( JIFIX(FTK) + ITIK(1) - 1 )/ITIK(1)
      CALL YTICKS( ITK )

```

C DETERMINE TICK INTERVALS FOR X AXIS AND SET THEM

```

      DS = ( SND - S1 )/FKA
      CALL PLPAR( DS, FM, ITK )
      FTK = ITK

```

```

      CALL GRAF( XMIN, FTK, XMAX, YMIN, YMAX-YMIN, YMAX )
      CALL FRAME

```

C PRINT TITLE FROM RUN CARD

```

C      CALL ANGLE( 90. )
C      CALL MESSAG( ITITLE, 60, -.3, 0. )
C      CALL ANGLE( -95. )

```

C*****

C DRAW BEAM SYSTEM

```

      S = S1
      call grace(.8)
      DO 140 K=N1,NBE
      NN = MI(K)

```

C FIND NAME AND OPNAME OF BEAM ELEMENT

```

      NAMK = MNAME(NN)
      NAMOP = INFF(1,NN)
      CALL DATA(NN,5,1,1,EL)
      SE = S + EL
      IGO=110

```

```

105 DO 106 JK=1,8
    IF (NAMOP.EQ.MOP(JK)) GO TO 107

```

```

106 CONTINUE
    GO TO 109

```

```

107 JN = JK
    GO TO (110,116), IGO

```

```

110 GO TO (135,117,117,112,112,112,112,112), JN

```

C IF MAGS.REF,INV.REF,EQU REFER TO MATRIX DEFINED BY A MAGNET, GET IT.

C OTHERWISE TREAT IT AS A COMPOSITE.

```

112 NN = MDAT(NN,1)
      NAMOP = INFF(1,NN)
      IGO=116
      GO TO 105

```

```

116 GO TO (135,117,117,109,109,109,109,109), JN

```

```

C      COMPOSITE. DRAW BOX LIKE BENDING MAGNET, BUT MUCH SMALLER.
109  YINC = .100/15.00
      PT = YMAX + YMAX*YINC
      PB = YMAX - YMAX*YINC
      GO TO 125
117  CONTINUE

C      DRAW MAGNET ABOVE OR BELOW THE LINE FOR FOCUSSED OF
C      DEFOCUSSED GRADIENT
C      STRADDLE THE LINE IF ZERO GRADIENT
      S = FDATA(NN,2)
      IF (G) 113,115,120
C      DEFOCUSSED
113  YINC = .700/15.00
      PT = YMAX
      PB = YMAX - YMAX*YINC
      GO TO 121
C      FOCUSSED
120  YINC = .700/15.00
      PT = YMAX + YMAX*YINC
      PB = YMAX
121  CONTINUE
      GO TO 125
C      ZERO GRADIENT
115  YINC = .3500/15.00
      PT = YMAX + YMAX*YINC
      PB = YMAX - YMAX*YINC

125  CONTINUE
C      DRAW MAGNETS
      X(1) = S
      Y(1) = PT
      X(2) = S
      Y(2) = PB
      X(3) = SE
      Y(3) = PB
127  X(4) = SE
      Y(4) = PT
      X(5) = S
      Y(5) = PT
      IF (K.EQ.N1) CALL GRACE( 8 )
      CALL CURVE( X, Y, 5, 0 )
      IF (K.EQ.NBE) CALL RESET( 'GRACE' )

135  CONTINUE
C      POSITION PAPER COORDINATES TO PRINT NAME AT TOP
C      WRITE (99,2), NAME
C      CENTER POINT
C      CENT = S + EL/2
C      CENTER SECOND LETTER OF ELEMENT
      S = SE
140  CONTINUE
      CALL RESET( 'GRACE' )
      GOTO 1

```

C*****

C FIND COORDINATES OF MAXIMUM POINTS

```

M1 = J1
M2 = J1
M3 = J1
M4 = J1
JJ = J1 + 1
Y1MX = Y1(J1)
Y2MX = Y2(J1)
Y3MX = Y3(J1)
Y4MX = Y4(J1)
JN = J1 + NN - 1
DO 184 J=JJ,JN
  IF (Y1(J).LE.Y1MX) GO TO 178
  Y1MX = Y1(J)
  M1 = J
178 IF (Y2(J).LE.Y2MX) GO TO 179
  Y2MX = Y2(J)
  M2 = J
179 IF (Y3(J).LE.Y3MX) GO TO 182
  Y3MX = Y3(J)
  M3 = J
182 IF (Y4(J).LE.Y4MX) GO TO 184
  Y4MX = Y4(J)
  M4 = J
184 CONTINUE

```

C*****

C SET UP PLOT

```

NN = NP
DO 172 I=1,KW
172 XPL(I)=XX(I)

```

C*** PLOT CURVES

C WRITE NAMES OF PLOTS AT MAXIMUM POINTS

C*****

C PLOT BETAX

```

DO 173 I=1,KW
  YPL(I)=Y1(I)
173 continue
  m1=j1
  do 1174 j=j1,j1+nn-1
    if(ypl(j).gt.ypl(m1)) m1=j
1174 continue
  CALL DASH
  CALL SETCLR('RED')
  CALL CURVE( XPL(J1), YPL(J1), NN, @ )

c CALL HEIGHT( .08 )
  PY = Y1(M1)
  PX = XX(M1)

```



```

      IF ( M1 .EQ. J1 ) PX = PX + .08
      IF ( M1 .EQ. JN ) PX = PX - .1
      IF (KB.NE.0) GO TO 181
      call rlmess('BX$',100,px,py)
      GO TO 183
181  CONTINUE
      CALL RLMESS( 'BX$', 100, PX-.05, PY+.07 )
183  CONTINUE
C          PLOT BETAY
      DO 174 I=1,KW
174  YPL(I)=Y3(I)
      CALL CHNDSH
      CALL SETCLR('BLUE')
      CALL CURVE( XPL(J1), YPL(J1), NN, 0 )

      PY = Y3(M3)
      PX = XX(M3)
      IF (M3.EQ.J1) PX = PX + .08
      IF (M3.EQ.JN) PX = PX - .10
      IF (KB.NE.0) GO TO 186
      call rlmess('BY$',100,px,py)
      GO TO 187
186  CONTINUE
      CALL RLMESS( 'BY$', 100, PX-.05, PY+.07 )
187  CONTINUE
c          CALL RESET( 'HEIGHT' )

C          PLOT XEQ,YEQ
      DO 188 I=1,KW
188  YPL(I)=Y2(I)
      FTK = FLT(2)
      ITK = ( JIFIX(FTK) + ITIK(2) - 1 )/ITIK(2)
      CALL RESET( 'SETCLR' )
      CALL YTICKS( ITK )
      CALL YGRAXS( PMX(3), PMX(2)-PMX(3), PMX(2), YAXIS, 'D$',
2          -100, XAXIS, 0. )
      CALL CHNDOT
      CALL SETCLR('GREEN')
      CALL CURVE( XPL(J1), YPL(J1), NN, 0 )

c          CALL HEIGHT( .08 )
      PY = Y2(M2)
      PX = XX(M2)
      IF (M2.EQ.J1) PX = PX + .08
      IF (M2.EQ.JN) PX = PX - .15
      CALL RLMESS( 'DX$', 100, PX, PY )

      DO 189 I=1,KW
189  YPL(I)=Y4(I)
      CALL RESET( 'SETCLR' )
      CALL CURVE( XPL(J1), YPL(J1), NN, 0 )

      PY = Y4(M4)
      PX = XX(M4)
      IF (M4.EQ.J1) PX = PX + .08

```

```

      IF (M4.EQ.JN) PX = PX - .15
      CALL RLMES( 'DY$', 100, PX, PY )
c     CALL RESET( 'HEIGHT' )
      CALL ENDPL(0)
200 CONTINUE
C      IF PLOTTED W, RETURN ARRAYS TO BETA
      IF (KB.NE.0) RETURN
      DO 202 I=1,KW
      Y1(I) = Y1(I)*Y1(I)
      Y3(I) = Y3(I)*Y3(I)
202 CONTINUE
      RETURN
C      ERROR IN INPUT PLOT POSITIONS
205 PRINT 10,IPRS
1     FORMAT (8A10)
2     FORMAT (A4)
3     FORMAT (F8.2)
4     FORMAT (6A10)
8     FORMAT (I3)
9     FORMAT (I5)
10    FORMAT (5X,'ERROR.  CANNOT PLOT FROM POSITION ',I5,' TO POSITION',
1     I5)
      RETURN
      END
C*****pltinit*****
      subroutine pltinit(mode,iuniti,iunito,iunitp)
c-----
c
c     ROUTINE TO INITIALIZE PLOTTING DEVICE AND PARAMETERS.
c
c-----
c
      character*8 dlast,devmy,devs(10)
      data ndevs,ldev/10,6/
      data dlast/'      '/
      data devs/ '4014  ','4107  ','4105  ','MAC    ',
-             'VT240  ','ZETA   ','mac    ','vt240  ',
-             'zeta   ','      '/
      if(mode.eq.'ZETA') go to 160
5 continue
      write(iunito,3)
3 format(/' Enter Device'/)
      read(iuniti,'(a)') devmy
4 continue
      do 10 i=1,ndevs
      if(devmy.eq.devs(i)) go to 20
10 continue
12 continue
      write(iunito,15) (devs(i),i=1,ldev)
15 format(/' Acceptable device names are'/(28x,a))
      go to 5
20 continue
      go to (110,120,130,140,150,160,140,150,160,180), i
110 continue
      call tekall(4014,960,1,0,0)

```

```

        go to 200
120 call tk41(4107)
        go to 200
130 call tk41(4105)
        go to 200
140 call tekall(4014,960,0.0,0)
        go to 200
150 call vt240
        go to 200
160 call zeta(1453,11,iunitp)
        go to 200
180 continue
        devmy=dlast
        go to 4
200 continue
        dlast=devmy
c
c — SET UNIT NUMBERS FOR ERROR AND SUMMARY MESSAGES AND DISALLOW
c BORDER AROUND THE PLOT
c
        call setdev( 50, 51 )
        call nobrdr
        return
        end
C*****PLTEND*****
        SUBROUTINE PLTEND

C—————
C
C ROUTINE TO TERMINATE PLOTTING
C—————
C
        CALL DONEPL

        RETURN
        END
C ***** PLTCON *****
        SUBROUTINE PLTCON(NSPT,SPT)
        IMPLICIT REAL*8 (A-H,O-Z)
CC      PLOTS POINTS IN ARRAY SPT ONTO ONE PAGE WITH LABELLED AXES
CC      NSPT IS THE NUMBER OF POINTS TO BE PLOTTED
CC      SPT(1,I) = X-COORD. OF ITH POINT
CC      SPT(2,I) = Y-COORD. OF ITH POINT
CC      SPT(3,I) = CHARACTER TO BE PLOTTED AT THIS POSITION
CC
        DIMENSION SPT(3,2) , ALINE(100), SAV(3)
        COMMON /UNIT / ISYSRD ,ISYSWR ,ISYSPU
        DIMENSION FMT(6), FMTIN(12)
        DATA DOT,SLASH,AI,ABLANK/1H., 1H/, 1HI, 1H /
        DATA MAXNX,MAXNY/ 98, 56/
        DATA FMT /4H(5X,,6HF15.7,,1H ,6HF15.7,,1H ,6HF15.7)/
        DATA FMTIN /1H , 4H 5X,, 4H10X,, 4H15X,, 4H20X,,
        * 4H25X,, 4H30X,, 4H35X,, 4H40X,, 4H45X,, 4H50X,, 4H55X,/
        IF (NSPT .LE. 1) RETURN
        XBEST = SPT(1,1)

```

```

YBEST = SPT(2,1)
CHBEST = SPT(3,1)
C      ORDER THE POINTS BY DECREASING Y
KM1 = NSPT - 1
DO 150 I= 1, KM1
  IQUIT = 0
  NI = NSPT - I
  DO 140 J= 1, NI
    IF (SPT(2,J) - SPT(2,J+1)) 120, 140, 140
120  DO 130 I3= 1, 3
    SAV(I3) = SPT(I3,J)
    SPT(I3,J) = SPT(I3,J+1)
130  SPT(I3,J+1) = SAV(I3)
    IQUIT = 1
140 CONTINUE
    IF (IQUIT) 160, 160, 150
150 CONTINUE
160 CONTINUE
C      FIND EXTREME VALUES
XMAX = SPT(1,1)
XMIN = XMAX
DO 200 I= 1, NSPT
  IF (SPT(1,I) .GT. XMAX) XMAX = SPT(1,I)
  IF (SPT(1,I) .LT. XMIN) XMIN = SPT(1,I)
200 CONTINUE
CALL BINSIZ(XMIN,XMAX,MAXNX,XMIN,XMAX,NX,BWIDX)
ANX = NX
YMAX = SPT(2,1)
YMIN = SPT(2,NSPT)
IF (YMAX .EQ. YBEST) YMAX=YBEST + 1.0D0
CALL BINSIZ(YMIN,YMAX,MAXNY,YMIN,YMAX,NY,BWIDY)
ANY = NY
IF (CHBEST .EQ. ABLANK) GO TO 50
XBEST = 0.5D0 * (XMAX+XMIN)
YBEST = 0.5D0 * (YMAX+YMIN)
50 CONTINUE
C      FIND SCALE CONSTANTS
AX = 1.0D0/BWIDX
AY = 1.0D0/BWIDY
BX = -AX*XMIN + 2.0D0
BY = -AY*YMIN - 2.0D0
C      CONVERT POINTS TO GRID POSITIONS
DO 300 I= 1, NSPT
  SPT(1,I) = AX*SPT(1,I) + BX
300 SPT(2,I) = ANY-AY*SPT(2,I) - BY
  NXBEST = AX*XBEST + BX
  NYBEST = ANY - AY*YBEST - BY
C      PRINT THE CONTOURS
NY = ANY + 1.0D0
NX = ANX + 2.0D0
ISP1 = 1
LINODD = 1
DO 400 I= 1, NY
  DO 310 IBK= 1, NX
310 ALINE (IBK) = ABLANK

```

```

      ALINE(1) = DOT
      ALINE(NX) = DOT
      ALINE(NXBEST) = DOT
      IF (I.NE.1 .AND. I.NE.NXBEST .AND. I.NE.NY) GO TO 320
      DO 315 J= 1, NX
315  ALINE(J) = DOT
320  CONTINUE
      YPRT = YMAX - DFLOAT(I-1)*BWIDY
      IF (ISP1 .GT. NSPT) GO TO 350
      DO 340 K= ISP1,NSPT
      KS = SPT(2,K)
      IF (KS .GT. I) GO TO 345
      IX = SPT(1,K)
340  ALINE(IX) = SPT(3,K)
      ISP1 = NSPT + 1
      GO TO 350
345  ISP1 = K
350  CONTINUE
      IF (LINODD .EQ. 1 .OR. I .EQ. NY) GO TO 380
      LINODD = 1
      WRITE (ISYSWR, 1007)      (ALINE(IW),IW=1,NX)
      GO TO 400
380  WRITE (ISYSWR, 1001) YPRT, (ALINE(IW),IW=1,NX)
      LINODD = 0
400  CONTINUE
      DO 410 IBK= 1, NX
410  ALINE (IBK) = ABLANK
      ALINE(1) = SLASH
      ALINE(NXBEST) = AI
      ALINE(NX-1) = SLASH
      WRITE (ISYSWR, 1007)      (ALINE(IW),IW=1,NX)
      N1 = NXBEST/5 - 1
      IF (N1 .LT. 1) N1 = 1
      IF (N1 .GT. 12) N1 = 12
      N2 = NX/5 - N1 - 3
      IF (N2 .GT. 0) GO TO 420
      IF (N1 .GT. 1) N1= N1-1
      N2 = 1
420  IF (N2 .GT. 12) N2 = 12
      FMT(3) = FMTIN(N1)
      FMT(5) = FMTIN(N2)
      WRITE (ISYSWR,FMT ) XMIN,XBEST,XMAX
      WRITE (ISYSWR,1002) BWIDX
500  RETURN
1001 FORMAT (F15.7,3H ... ,100A1)
1002 FORMAT (40X,13H ONE COLUMN = F13.7)
1003 FORMAT (5X A10)
1007 FORMAT (18X , 100A1)
      END
C ***** PLPAR *****
      SUBROUTINE PLPAR(PM,FL,IT)
      IMPLICIT REAL*8 (A-H,O-Z)
C      ROUND LIMITS FOR PLOT OF BETATRON FUNCTIONS.  SET TICK INTERVALS.

```

```
    IF (PM.GT.5.) GO TO 43
    FL = 5.
    IT = 1
    RETURN
43  IF (PM.GT.10.) GO TO 44
    FL = 10.
    IT = 2
    RETURN
44  IF (PM.GT.25.) GO TO 45
    FL = 25.
    IT = 5
    RETURN
45  IF (PM.GT.50.) GO TO 46
    FL = 50.
    IT = 10
    RETURN
46  IF (PM.GT.100.) GO TO 471
    FL = 100.
    IT = 20
    RETURN
471 IF (PM.GT.150.) GO TO 47
    FL = 150.
    IT = 30
    RETURN
47  IF (PM.GT.250.) GO TO 48
    FL = 250.
    IT = 50
    RETURN
48  IF (PM.GT.500.) GO TO 49
    FL = 500.
    IT = 100
    RETURN
49  IF (PM.GT.1000.) GO TO 50
    FL = 1000.
    IT = 200
    RETURN
50  IF (PM.GT.1500.) GO TO 51
    FL = 1500.
    IT = 300
    RETURN
51  IF (PM.GT.2500.) GO TO 52
    FL = 2500.
    IT = 500
    RETURN
52  IF (PM.GT.5000.) GO TO 53
    FL = 5000.
    IT = 1000
    RETURN
53  IP = PM/1000.
    FL = (IP+1)*1000
    IT = 2000
    RETURN
END
```

```
COMMON/ TRKINT/G,EX,EY,OMSQ,SEND,VMX,VMN,DINT,BEG,DPR,JPR,
1  THET,PLT,NPT,NZ,PMAX,NSIZ,ITITLE(15)
LOGICAL BEG,DPR,PLT
```

File: TSW.CCC;1

```
COMMON/TSW/TRSW,MCY(20),JM,LOCC
LOGICAL TRSW
```

File: VAX.COM;2

To prepare a VAX version of "SYNCH":
 Compile the following from BNLDAG::DUA0:[parsat.synch.synvax87]
 MAINS.FOR (main program)
 SYNI.FOR (subroutines with names from A to G)
 SYNII.FOR (subroutines with names from G to P)
 REVMAT.FOR (a recently corrected version of one subroutine)
 SYNNOPL.FOR (subroutines with names from P to Z)
 (alternate: SYNPL.FOR includes references to LBL DISSPLA system)
 All the *.CCC files (COMMON files) must be present in the directory for
 compilation.
 Then link resulting OBJ programs into EXE ***
 LINK/EXE=SYNCH MAINS,SYNI,SYNII,REVMAT,SYNNOPL

To run "Synch": program looks for data in FOR002.Dat; main output is in
 FOR003.Dat; auxiliary output in various other files.

File: VERSION.CCC;1

```
COMMON/VERSION/CODE(2),ID,VER
```

File: WRK.CCC;1

```
COMMON/WRK/BZ,BQ,AZ,DPSI,SM,SL,PSI,WW,WT,BETW,P,Q,
1  T,U,BKZ,BKZP,CZ,SZ,CHZB,SHZB,PHI,BK,SA,BL,BM,
2  AQ,RHO,BETS,PSIWS,BKZI,BKZPI,G(6),AB,C,S,CHB,
3  SHB,E,EI,SZCB,SZBC,CZBC,CZCB,SZSB,SZRS,CZBS,CZSB,
4  ZW,ZT,BKI,Y,Z,Y1,Y2,Y3,Y4
```

C

File: XXYY.CCC;1

```
COMMON /XXYY/ XX(4000),Y1(4000),Y2(4000),Y3(4000),Y4(4000)
```

END