Computer Physics Communications 39 (1986) 173–180
North-Holland, Amsterdam

# FORTRAN CODE FOR THE THREE-DIMENSIONAL ISING MODEL

Michael CREUTZ

*Department of Physics, Brookhaven National Laboratory, Upton, NY 11973, USA*

and

K.J.M. MORIARTY

*Institute for Computational Studies, Department of Mathematics, Statistics and Computing Science, Dalhousie University, Halifax, Nova Scotia B3H 4H8, Canada*

Received 25 April 1985; in revised form 15 August 1985

## PROGRAM SUMMARY

*Title of program*: MICROIS

*Catalogue number*: AADW

*Program available from*: CPC Program Library, Queen's University of Belfast, N. Ireland (see application form in this issue)

*Computer*: CDC CYBER 170-730 (dual processor); *Installation*: Dalhousie University Computer Center

*Operating system*: CDC NOS 2.2

*Programming language used*: FORTRAN-77

*High speed storage required*: 25 Kwords

*Number of bits in a word*: 60

*Peripherals used*: terminal, line printer

*Number of lines in combined program and test deck*: 452

*Keywords*: solid state Ising model, phase transitions, critical exponents, correlation functions, magnetization, microcanonical methods

*Nature of the physical problem*
The program calculates equilibrium distributions for the elementary magnets in a large three-dimensional Ising model system so that the critical inverse temperature and critical exponents can be measured.

*Method of solution*
An essentially deterministic technique called the microcanonical method with demons [1] is used to obtain equilibriated elementary magnetic configurations. Spin correlations at a variety of separations can be measured.

*Restrictions on the complexity of the program*
In order to reduce the errors on our measurements, reasonably long run times are required which is the only restriction on the use of the program. Primary memory is not a restriction as very little memory is required.

*Typical running time*
The test run on an $8^2 \times 120$ lattice with 10 iterations through the lattice, each with 10 sweeps, took 3.2 s on the CDC CYBER 170-730.

*Reference*
[1] M. Creutz, Phys. Rev. Lett. 50 (1983) 1411.
G. Bhanot, M. Creutz and H. Neuberger, Nucl. Phys. B235 [FS11] (1984) 417.

## LONG WRITE-UP

## 1. Introduction

The Ising model is one of the simplest theoretical models constructed in an attempt to simulate the phenomenon of phase transitions in general and the ferromagnetic phase transition in particular. This model can be solved analytically in two dimensions [1] and exhibits the kind of behavior shown by ferromagnets: viz., above a certain temperature, called the critical temperature, the elementary magnets in a large sample tend to be in a disordered state, so that there is no net magnetization, whereas, below that temperature, the magnets tend to order themselves parallel to one another, with the effect that there is a net magnetization. The more realistic three-dimensional Ising model has so far eluded analytic solution, though various parameters have been estimated by approximation techniques. Some of these approximation methods are analytical and can be implemented by hand, but nowadays the trend is to simulate the model on a computer and thereby extract its properties. As these properties can be more reliably determined for larger systems, i.e., for larger numbers of interacting elementary magnets, it is obvious that for better results there is a need for larger computation times (and greater memory).

We work with the Ising model in 3 dimensions. On any site $i$ of a 3-dimensional lattice is a spin variable $s_i$ which takes values from the set $\{1, -1\}$. The interaction of these spins is given by the Hamiltonian

$$H = \sum_{\{i,j\}} s_i s_j,$$

where $\{i,j\}$ denotes the set of all nearest neighbor pairs of sites. The program releases the demons, each of which carry small amounts of energy around the lattice. As with the usual Monte Carlo simulations, the path through the lattice is arbitrary. We hop through the lattice in an orderly manner with big steps of a few tens of sites. Upon reaching a site a demon attempts to flip the corresponding spin. If the change in the spin energy can be absorbed by the demon, it accepts the flip. If, on the other hand, the resulting change in the demon energy would take it out of the allowed range, the spin flip is rejected.

What are the quantities that we are measuring? There are various parameters associated with the ferromagnetic phase transition, not all of equal importance. The critical temperature itself, as well as the value of the internal energy at this temperature are relatively unimportant quantities because they are specific to the model and are not related to any physical systems. They are nevertheless of some interest because they have been estimated by other methods, so that a comparison can be made. Moreover, they are needed for calculating some of the more important quantities. These are the critical exponents. A full discussion of these can be found in ref. [2].

In a previous paper [3], a program for carrying out three-dimensional Ising model calculations with the microcanonical method with demons [4] was presented. However, this program was very machine dependent being partly written in the CDC assembly language code COMPASS. Thus, the program was not transportable. After some encouragement from our colleagues, we wish to present a fully transportable code written in standard ANSI FORTRAN-77. A discussion of this code will be the object of the present paper.

## 2. Code description

The program consists of the following routines: MICROIS (main program), BETA, ENERGY, AMIX, IBCOUNT, MONTE, CORX and CORZ (see the flow chart in fig. 1). These routines have the following functions:

1) MICROIS is the main driver routine. This routine initializes the program parameters such as the lattice size and the total energy per bond, initiates the simulation and measuring procedures.
2) BETA finds the inverse temperature $\beta$ given an average demon energy E.
3) ENERGY measures the lattice energy.
4) AMIX randomly permutes bits.

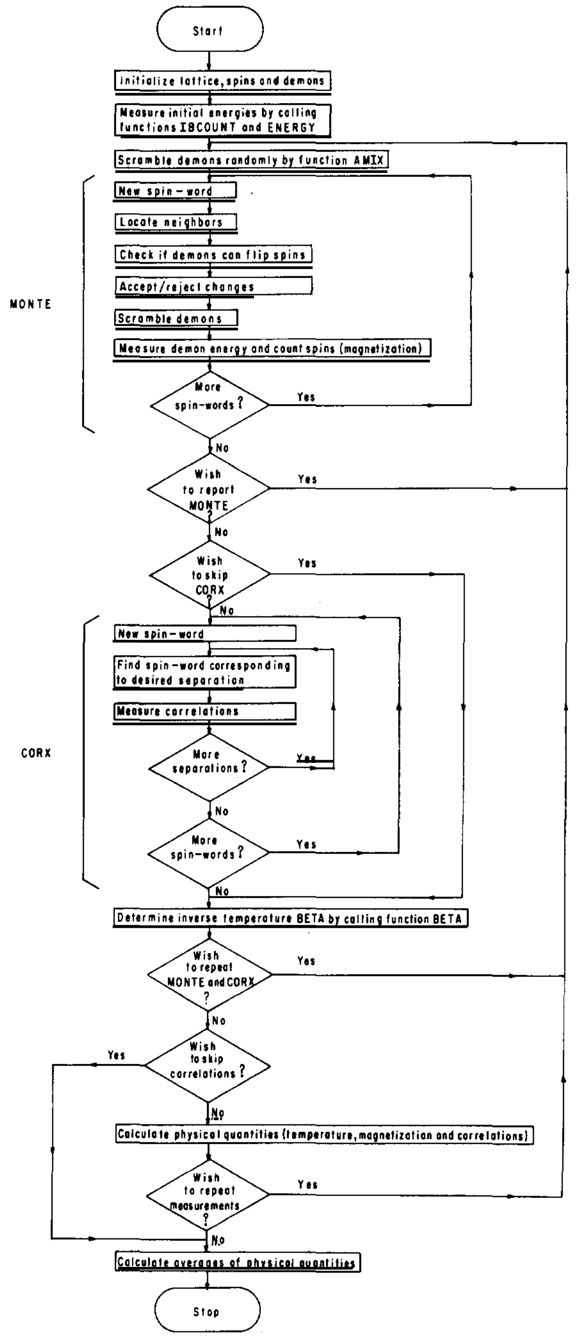
Start
Initialize lattice, spins and demons
Measure initial energies by calling functions IBCOUNT and ENERGY
Scramble demons randomly by function AMIX
New spin-word
Locate neighbors
Check if demons can flip spins
Accept/reject changes
Scramble demons
Measure demon energy and count spins (magnetization)
More spin-words ?
Yes
No
MONTE
Wish to report MONTE ?
Yes
No
Wish to skip CORX ?
Yes
No
New spin-word
Find spin-word corresponding to desired separation
Measure correlations
More separations ?
Yes
No
More spin-words ?
Yes
No
CORX
Determine inverse temperature BETA by calling function BETA
Wish to repeat MONTE and CORX ?
Yes
No
Wish to skip correlations ?
Yes
No
Calculate physical quantities (temperature, magnetization and correlations)
Wish to repeat measurements ?
Yes
No
Calculate averages of physical quantities
Stop

Fig. 1. Schematic flow chart of the three-dimensional Ising model program.

5) IBCOUNT counts the bits that are set in the bit string Y.
6) MONTE carries out one sweep over the lattice. The demon energy and magnetization are measured.
7) CORX counts the antiparallel spins separated by N sites in the $x$-direction.
8) CORZ counts the antiparallel spins separated by N sites in the $z$-direction.

A listing of the code, together with detailed comments, is presented at the end of the paper. A brief description of these routines was given in ref. [3].

The CDC CYBER 170-730, on which the code was developed, is a 60-bit word length machine. The industrial standard is now words of 64 bits. To convert the code at the end of the paper from a 60-bit to a 64-bit word length, the user must make the following alterations:

| | | |
|---|---|---|
| line 22 | 120 | becomes 128 |
| line 49 | 45 | becomes 48 |
| line 50 | 120 | becomes 128 |
| line 52 | 30 | becomes 32 |
| line 57 | 60 | becomes 64 |
| line 62 | 59 | becomes 63 |
| line 72 | 45 | becomes 48 |
| line 77 | 120 | becomes 128 |
| line 141 | 60 | becomes 64 |
| line 146 | 60 | becomes 64 |
| line 147 | 30 | becomes 32 |
| line 159 | 30 | becomes 32 |
| line 265 | 59 | becomes 63 |
| line 282 | 180 | becomes 192 |
| line 295 | 30 | becomes 32 |
| line 300 | 60 | becomes 64 |
| | 60 | becomes 64 |
| line 319 | 15 | becomes 16 |
| line 322 | 15 | becomes 16 |
| line 323 | 30 | becomes 32 |
| | 45 | becomes 48 |
| line 373 | 58 | becomes 62 |
| line 383 | 23 | becomes 27 |

Subroutine IBCOUNT is a very compute intensive part of the code being called very often. To obtain a really efficient code on a scalar computer, this routine should be written in assembly language. A COMPASS version of this subroutine is contained in ref. [2]. Some alternate FORTRAN versions can be found in the appendix.

Some preliminary results of possible measure-

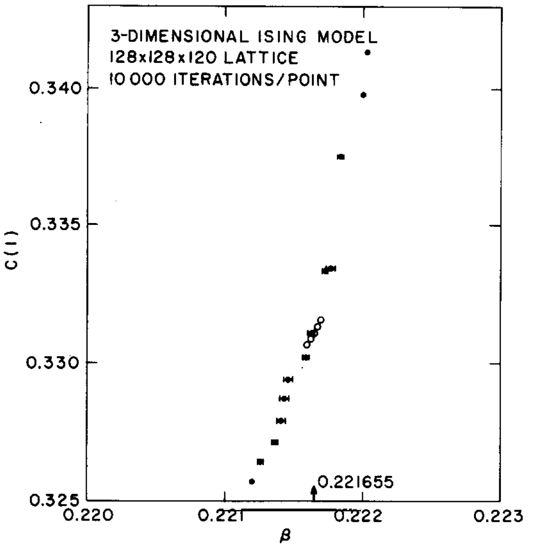

Fig. 2. The nearest neighbor correlation in the three-dimensional Ising model ($128\times128\times120$ lattice) near the critical inverse temperature as a function of the inverse temperature. The open circles represent the results of the Santa Barbara Ising Model machine [5] using a $64^3$ lattice.

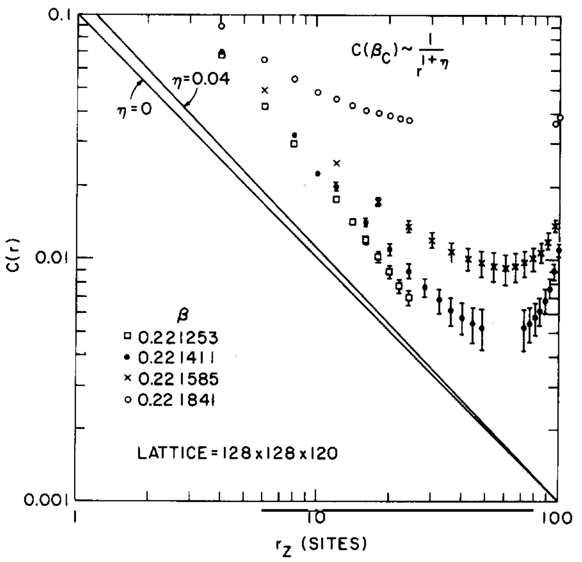

Fig. 3. The correlation functions for separation in the three-dimensional Ising model ($128\times128\times120$ lattice) near the critical inverse temperature as a function of the site separation $r_z$.

ments of the three-dimensional Ising model on a $128\times128\times120$ lattice at the critical inverse temperature $\beta_c = 0.2217$ are shown in figs. 2–4. The results were obtained on a CDC 7600. These results are included to indicate the potential of the method. In fig. 2 we present data on the nearest neighbor correlation near the critical inverse temperature. Each data point is the result of 10000 iterations through the lattice with the averaging procedure carried out over the last 9000 iterations. The error bars are in the inverse temperature $\beta$ because of the use of the microcanonical method. Fig. 3 shows the correlation of spins near the critical inverse temperature. At the critical inverse temperature $\beta_c$, this is expected to behave like

$$C(\beta_c) \sim 1/r^{1+\eta},$$

where $\eta$ is a characteristic parameter of the model. The results for $\eta = 0.0$ and 0.04 are shown in fig. 3. The ratios of correlation functions near the critical inverse temperature are shown in fig. 4. The value corresponding to $\eta = 0.04$ is also shown.

Our present plans involve running our program for anything up to 1 M iterations through large lattices in order to obtain high-grade statistics and thus allow us to make accurate measurements of the critical inverse temperature and the critical exponents. Results on this work will be reported shortly.

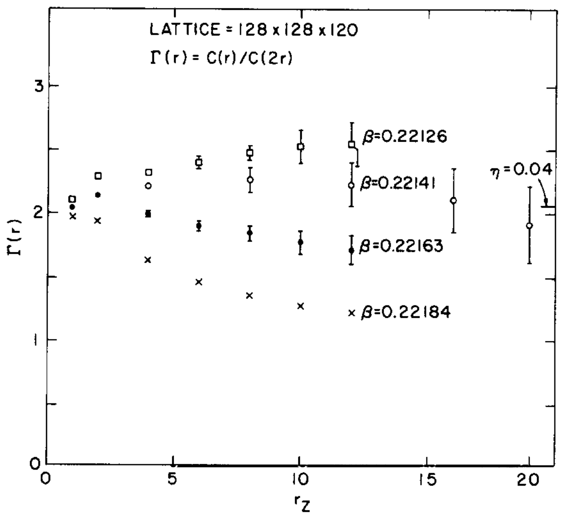

Fig. 4. The ratio of correlation functions in the three-dimensional Ising model ($128\times128\times120$ lattice) near the critical inverse temperature for separations $r$ and $2r$ as a function of site separation $r_z$.

For an alternate approach to the 3-dimensional Ising model, see ref. [6].

## Acknowledgements

We would like to thank P. Mitra and M. O'Brien for useful advice in setting up this code. We would also like to thank Dalhousie University Computer Center for the granting of time on their CDC CYBER 170-730 where this calculation was undertaken. This research was also carried out in part under the auspices of the US Department of Energy under contract No. DE-AC02-76CH00016 and the Natural Science and Engineering Research Council of Canada under grant NSERC A8420.

## Appendix

For completeness, we give an alternate but slower version of the subroutine IBCOUNT(Y), which is

```
*     FUNCTION IBCOUNT(Y)                                        0304
      -------------------                                        0305
                                                                 0306
*     FUNCTION IBCOUNT(Y) COUNTS SET BITS IN Y                   0307
      N=1                                                        0308
      IBCOUNT=0                                                  0309
      X=MASK(30)                                                 0310
      I=SHIFT(AND(Y,X),30)                                       0311
    1 IF(I.LE.1)GOTO 3                                           0312
    2 J=SHIFT(I,-1)                                              0313
      IBCOUNT=IBCOUNT+I-J-J                                      0314
      I=J                                                        0315
      IF(I.GT.1)GOTO 2                                           0316
    3 IBCOUNT=IBCOUNT+I                                          0317
      IF(N.LT.0)GOTO 4                                           0318
      I=AND(Y,COMPL(X))                                          0319
      N=-1                                                       0320
      GOTO 1                                                     0321
    4 RETURN                                                     0322
      END                                                        0323
                                                                 0324
```

where, when going from a 60-bit to a 64-bit word length machine, the 30 in lines 310 and 311 would become 32. Another version would replace the body of this routine by the code

```
  IBCOUNT = 0
  DO 1 N = 1,60
  IBCOUNT = IBCOUNT + AND(1, Y)
1 Y = SHIFT(Y, 1)
  RETURN
  END.
```

For a 64-bit word length machine, the loop control variable N goes from 1 to 64.

## References

[1] L. Onsager, Phys. Rev. 65 (1944) 117.
T.D. Schultz, D.C. Mattis and E.H. Lieb, Rev. Mod. Phys. 36 (1964) 856.

[2] M. Creutz, P. Mitra and K.J.M. Moriarty, Computer Investigations of Three-Dimensional Ising Model, Brookhaven National Laboratory Preprint BNL-34741; J. Stat. Phys. (to be published).

[3] M. Creutz, P. Mitra and K.J.M. Moriarty, Comput. Phys. Commun. 33 (1984) 361.

[4] M. Creutz, Phys. Rev. Lett. 50 (1983) 1411.
G. Bhanot, M. Creutz and H. Neuberger, Nucl. Phys. B235 [FS11] (1984) 417.

[5] R.B. Pearson, J.L. Richardson and D. Toussaint, J. Comput. Phys. 51 (1983) 241.
M.N. Barber, R.B. Pearson, D. Toussaint and J.L. Richardson, Santa Barbara report NSF-ITP-83-144 (1983).

[6] G.S. Pawley, R.H. Swendsen, D.J. Wallace and K.G. Wilson, Phys. Rev. B29 (1984) 4030.

## PROGRAM LISTING

```
*     PROGRAM MICROIS
      ---------------
*     THIS IS A FORTRAN PROGRAM FOR STUDYING
*     CORRELATION FUNCTIONS OF THE THREE-DIMENSIONAL
*     ISING MODEL ON A SIMPLE CUBIC LATTICE
*     BY THE MICROCANONICAL METHOD
*     FOR DETAILED INFORMATION AND REFERENCES SEE THE PAPER
*     "FORTRAN CODE FOR THE THREE-DIMENSIONAL
*     ISING MODEL"
*     BY M. CREUTZ, DEPARTMENT OF PHYSICS,
*     BROOKHAVEN NATIONAL LABORATORY,
*     UPTON, LONG ISLAND, NEW YORK, 11973 AND
*     K.J.M. MORIARTY, INSTITUTE FOR COMPUTATIONAL
*     STUDIES, DEPARTMENT OF MATHEMATICS, STATISTICS
*     AND COMPUTING SCIENCE, DALHOUSIE
*     UNIVERSITY, HALIFAX, NOVA SCOTIA B3H 4H8,
*     CANADA

*               1. INITIALIZING THE PROGRAM
*
*     THE VARIABLES IX,IY DETERMINE THE LATTICE SIZE,
*     WHICH IS IX*IY*120
*     ARRAY SPIN SHOULD HAVE DIMENSION 2*IX*IY
*     NOTE THAT THIS HOLDS ALSO FOR FUNCTION ENERGY
*     AND SUBROUTINES MONTE,CORX,CORZ
*     THIS PROGRAM MEASURES CORRELATIONS FOR 3 SEPARATIONS
*     N(1),N(2),N(3) ARE TO BE SET TO THE DESIRED SEPARATIONS

*     CORRELATIONS IN X-DIRECTION ARE BEING MEASURED
*     CORRELATIONS IN Z-DIRECTION MAY BE MEASURED BY
*     USING SUBROUTINE CORZ PROVIDED(BUT NOT USED HERE)

      DIMENSION SPIN(128)
      DIMENSION N(3),IC(3),CS(3),CS2(3),ICORR(3),CORR(3)

      IX=IY=8

      N(1)=IX/4
      N(2)=IX/2
      N(3)=(IX+IY*IY)/2

      IMAX=2*IX*IY

*               2. INITIALIZING THE SPINS

*     PARAMETER E DETERMINES APPROX. TOTAL ENERGY PER BOND

      E=.655
      IE=2*INT(E*IMAX*45)
      KO=(IE-120-IMAX)/16
      DO 1 I=1,IMAX
    1 SPIN(I)=MASK(30)
      IM4=IMAX/4
      K1=KO/IM4
      DO 2 I=4,IMAX,4
      SPIN(I)=SHIFT(SPIN(I),K1)
    2 SPIN(I-3)=SHIFT(SPIN(I-3),60-K1)
      K2=4*MOD(KO,IM4)
      IF(K2.EQ.0)GOTO 4
      DO 3 I=4,K2,4
      SPIN(I)=SHIFT(SPIN(I),1)
    3 SPIN(I-3)=SHIFT(SPIN(I-3),59)
    4 NFLIP=2*MINO(2*IX,K2,IMAX-K2)+IMAX+16*(K1*IM4+K2/4)

*               3. INITIALIZING THE DEMONS

      DMN1=MASK((IE-NFLIP)/2)
      DMN2=0.

*               4. MEASURING INITIAL ENERGIES

      ED=IBCOUNT(DMN1)/(45.*IMAX)
      E=ENERGY(SPIN,IX,IMAX)
      ETOT=E+ED
      PRINT*
      PRINT*
      PRINT*,' LATTICE SIZE: ',IX,' X ',IY,' X 120'
      PRINT*
      PRINT*
      PRINT*,' ELATT= ',E,' EDEM= ',ED
      PRINT*,' ETOT= ',ETOT

*               5. SIMULATION

*     THREE ITERATION PARAMETERS ARE INVOLVED:
*     SWEEP NSWEEP TIMES,MEASURING DEMON ENERGY AND
*     MAGNETIZATION WITHIN MONTE(THIS CAN BE SUPPRESSED
*     BY REMOVING TWO LINES IN MONTE)
*     THEN MEASURE CORRELATIONS(EXCEPT AFTER 1ST SET OF SWEEPS)
*     CALCULATE AVERAGES AFTER NIT MEASUREMENTS
*     REPEAT NBATCH TIMES AND FIND GRAND AVERAGES

      NSWEEP=10
      NIT=10
      NBATCH=5

      ABATCH=1.*NBATCH-1.
      BS=BS2=AMS=AMS2=0.
      R1=R2=R1E=R2E=0.
      DO 5 I=1,3
    5 CS(I)=CS2(I)=0.
*     MAJOR LOOP STARTS

      DO 11 MEAS=1,NBATCH

      PRINT*
      PRINT*
      PRINT*,' AVERAGE OVER ',NIT,' ITERATIONS'
     +,',EACH WITH ',NSWEEP,' SWEEPS '

      IED=IS=0

      DO 6 I=1,3
    6 ICORR(I)=0

*     NOTE THE TIME -- MONITOR SPEED OF PROGRAM
      T=SECOND()

      DO 8 ITER=1,NIT

      DO 7 NSW=1,NSWEEP

*     SCRAMBLE DEMONS RANDOMLY
      DMN1=AMIX(DMN1)
      DMN2=AMIX(DMN2)

      CALL MONTE(DMN1,DMN2,SPIN,IX,IMAX,ISUM,IMAG)

      IED=IED+ISUM
    7 IS=IS+IMAG

      IF(MEAS.EQ.1)GOTO 8

      CALL CORX(SPIN,IMAX,N,IC)

      ICORR(1)=ICORR(1)+IC(1)
      ICORR(2)=ICORR(2)+IC(2)
      ICORR(3)=ICORR(3)+IC(3)
    8 CONTINUE

      RATE=IMAX*NIT*NSWEEP*60/(1.E6*(SECOND()-T))
      PRINT*,' RUNNING AT ',RATE,' MFLIPS '

*     CALCULATE AV. DEMON ENERGY AND MAGNETIZATION

      ED=IED/(60.*NSWEEP*NIT*IMAX)
      S=1.-IS/(30.*NSWEEP*NIT*IMAX)

*     DETERMINE BETA AND SISJ

      BET=BETA(ED)
      SISJ=1.-ETOT+4*ED/(3.*IMAX)
      PRINT*,' BETA= ',BET,' SISJ= ',SISJ
      PRINT*,' MAGNETIZATION= ',S
      IF (MEAS.EQ.1)GOTO 11
*     OTHERWISE CALCULATE CORRELATIONS

      DO 9 I=1,3
    9 CORR(I)=1.-ICORR(I)/(30.*NIT*IMAX)

      PRINT*,' CORR(IX/4)= ',CORR(1)
      PRINT*,' CORR(IX/2)= ',CORR(2)
      PRINT*,' CORR(IX/2,IY/2)= ',CORR(3)

*     ACCUMULATE RESULTS OF MEASUREMENT

      BS=BS+BET
      BS2=BS2+BET**2
      AMS=AMS+S
      AMS2=AMS2+S**2

      DO 10 M=1,3
      CS(M)=CS(M)+CORR(M)
   10 CS2(M)=CS2(M)+CORR(M)**2

*     CALCULATE RATIOS OF CORRELATIONS

      R1=R1+CORR(1)/CORR(2)
      R2=R2+CORR(2)/CORR(3)
      R1E=R1E+(CORR(1)/CORR(2))**2
      R2E=R2E+(CORR(2)/CORR(3))**2
   11 CONTINUE
*     MAJOR LOOP OVER

*               6. FINAL CALCULATIONS

      BS=BS/ABATCH
      BS2=SQRT((BS2/ABATCH-BS**2)/(ABATCH-1.))
      AMS=AMS/ABATCH
      AMS2=SQRT((AMS2/ABATCH-AMS**2)/(ABATCH-1.))

      PRINT*
      PRINT*,' *** AVERAGES AFTER DISCARDING FIRST BATCH *** '
      PRINT*,' AV. BETA= ',BS,' +/- ',BS2
      PRINT*,' AV. MAG.= ',AMS,' +/- ',AMS2

      DO 12 M=1,3
      CS(M)=CS(M)/ABATCH
   12 CS2(M)=SQRT((CS2(M)/ABATCH-CS(M)**2)/(ABATCH-1.))

      PRINT*,' AV. CORR(IX/4)= ',CS(1),' +/- ',CS2(1)
      PRINT*,' AV. CORR(IX/2)= ',CS(2),' +/- ',CS2(2)
      PRINT*,' AV. CORR(IX/2,IY/2)= ',CS(3),' +/- ',CS2(3)
      R1=R1/ABATCH
      R2=R2/ABATCH
      R1E=SQRT((R1E/ABATCH-R1**2)/(ABATCH-1.))
      R2E=SQRT((R2E/ABATCH-R2**2)/(ABATCH-1.))

      PRINT*,' CORR(IX/4)/CORR(IX/2)= ',R1,' +/- ',R1E
      PRINT*,' CORR(IX/2)/CORR(IX/2,IY/2)= ',R2,' +/- ',R2E
      PRINT*
      PRINT*

      STOP
      END





      FUNCTION BETA(E)
*     ----------------

*     FINDS BETA GIVING AVERAGE DEMON ENERGY E

      F(X)=(X+2*X**2+3*X**3)/(1+X+X**2+X**3)
      XL=0
      XH=1.5

      DO 1 N=1,50
      XN=(XL+XH)*.5
      FN=F(XN)
      IF (FN.GT.E)XH=XN
    1 IF (FN.LT.E)XL=XN

      BETA=-ALOG(XN)/4

      RETURN
      END




      FUNCTION ENERGY(SPIN,IX,IMAX)
*     -----------------------------

*     MEASURES LATTICE ENERGY

*     CHANGE DIMENSION OF SPIN IF CHANGE LATTICE SIZE

      DIMENSION SPIN(128),B(6)

      N1=3
      N2=2*IX+1
      N3=IMAX-2*IX+1
      N4=IMAX-1
      JSHIFT=1
      IE=0
      DO 2 NROW=1,IMAX
      AOLD=SPIN(NROW)
      B(1)=SPIN(N1)
      B(2)=SPIN(N2)
      B(3)=SPIN(N3)
      B(4)=SPIN(N4)
      B(5)=SPIN(NROW+JSHIFT)
      C=SHIFT(B(5),59)
      B(6)=SHIFT(C,JSHIFT+1)
      N1=N1+1
      N2=N2+1
      N3=N3+1
      N4=N4+1
      IF(N4.LE.IMAX)GOTO 1
      N5=N4
      N4=N3
      N3=N2
      N2=N1
      N1=N5-IMAX
    1 JSHIFT=-JSHIFT

      DO 2 NBR=1,6
    2 IE=IE+IBCOUNT(XOR(AOLD,B(NBR)))

      ENERGY= IE/(IMAX*180.)

      RETURN
      END




      FUNCTION AMIX(DMN)
*     ------------------

*     PERMUTES BITS SEMI-RANDOMLY

      L=INT(30*RANF())
      P1=MASK(L).AND.DMN
      P2=MASK(L).AND.SHIFT(DMN,L)
      P3=(.NOT.MASK(2*L)).AND.DMN

      AMIX=SHIFT(P2.OR.SHIFT(P1,60-L).OR.P3,INT(60*RANF()))

      RETURN
      END
```

```
      FUNCTION IBCOUNT(X)
*     -------------------

*     THIS FUNCTION RETURNS THE NUMBER OF SET
*     BITS IN WORD X
*
*     IT IS AWKWARD TO DO THIS IN FORTRAN; IT
*     WOULD BE BETTER TO WRITE AN EQUIVALENT
*     FUNCTION IN ASSEMBLY LANGUAGE
*
*     FOR A 64-BIT MACHINE, CHANGE 15 TO 16, 30 TO
*     32, 45 TO 48 AND CHANGE
*     1000010000100001OCTAL TO 1000100010001HEX
*
      IBCOUNT=0
      Y=X
      DO 1 N=1,15
      IBCOUNT=IBCOUNT+AND(O"1000010000100001",Y)
    1 Y=SHIFT(Y,1)
      IBCOUNT=AND((IBCOUNT+SHIFT(IBCOUNT,15)
     X+SHIFT(IBCOUNT,30)+SHIFT(IBCOUNT,45)),127)
      RETURN
      END




      SUBROUTINE MONTE(DMN1,DMN2,SPIN,IX,IMAX,ISUM,IMAG)
*     --------------------------------------------------

*     MONTE CARRIES OUT ONE SWEEP OVER THE LATTICE
*     DEMON ENERGY AND MAGNETIZATION ARE MEASURED IN LINES
*     ISUM=ISUM+... AND IMAG=IMAG+...
*     THESE LINES MAY BE DROPPED TO SPEED UP SIMULATION

      DIMENSION SPIN(128),SNBR(6),NBR(4)
*     CHANGE DIMENSION OF SPIN IF NECESSARY

      IHOP=11
*     THIS IS THE STEP SIZE OF DEMONS
*     IHOP AND IMAX MUST BE RELATIVELY PRIME

      ISUM=IMAG=0
      JSHIFT=1

      NROW=1

*     LOCATE NBRS IN X- AND Y-DIRECTIONS
      NBR(1)=3
      NBR(2)=2*IX+1
      NBR(3)=IMAX-2*IX+1
      NBR(4)=IMAX-1

*     START SIMULATION

    1 DP1=COMPL(DMN1)
      DP2=XOR(DMN1,DMN2)
      ACCEPT=AND(DMN1,DMN2)
      OLD=SPIN(NROW)

      DO 2 M=1,4
      SNBR(M)=SPIN(NBR(M))
    2 NBR(M)=NBR(M)+IHOP
      IF (NBR(4).LE.IMAX)GOTO 4
    3 NBRONE=NBR(4)
      NBR(4)=NBR(3)
      NBR(3)=NBR(2)
      NBR(2)=NBR(1)
      NBR(1)=NBRONE-IMAX
      IF(NBR(4).GT.IMAX)GOTO 3
    4 SNBR(5)=SPIN(NROW+JSHIFT)
      SNBR(6)=SHIFT(SNBR(5),1)
      IF(JSHIFT.EQ.-1)SNBR(6)=SHIFT(SNBR(6),58)
      JSHIFT=-JSHIFT
      DO 5 NEBR=1,6
      C=XOR(OLD,SNBR(NEBR))
      ACCEPT=XOR(ACCEPT,AND(C,DP1,DP2))
      DP2=XOR(DP2,AND(C,DP1))
    5 DP1=XOR(DP1,C)
*     ACCEPT CHANGES WHERE APPLICABLE AND SCRAMBLE DEMONS
      SPIN(NROW)=XOR(OLD,ACCEPT)
      DMN1=SHIFT(OR(AND(DMN1,COMPL(ACCEPT)),AND(DP1,ACCEPT)),37)
      DMN2=SHIFT(OR(AND(DMN2,COMPL(ACCEPT)),AND(DP2,ACCEPT)),23)

*     THE NEXT TWO LINES MEASURE DEMON ENERGY AND MAGNETIZATION
      ISUM=ISUM+IBCOUNT(DMN1)+2*IBCOUNT(DMN2)
      IMAG=IMAG+IBCOUNT(SPIN(NROW))

      NROW=NROW+IHOP
      IF(NROW.LE.IMAX)GOTO 1
      NROW=NROW-IMAX
      IF(NROW.NE.1)GOTO 1

      RETURN
      END
```

```
*     SUBROUTINE CORX(SPIN,IMAX,N,IC)
      -------------------------------
*     COUNTS ANTIPARALLEL SPINS SEPARATED BY N SITES
*     IN THE X DIRECTION AND PLACES RESULTS IN IC

*     IF N EXCEEDS IX,SEPARATION DEVELOPS COMPONENT IN Y DIRECTION

      DIMENSION SPIN(128),N(3),IC(3),NBR(3)
*     CHANGE DIMENSION OF SPIN IF REQUIRED

      IC(1)=IC(2)=IC(3)=0
      DO 1 NROW=1,IMAX
      DO 1 M=1,3
      NBR(M)=NROW+N(M)*2
      IF (NBR(M).GT.IMAX)NBR(M)=NBR(M)-IMAX
    1 IC(M)=IC(M)+IBCOUNT(XOR(SPIN(NROW),SPIN(NBR(M))))

      RETURN
      END




*     SUBROUTINE CORZ(SPIN,IMAX,N,IC)
      -------------------------------
*     COUNTS ANTIPARALLEL SPINS SEPARATED BY N SITES
*     IN THE Z DIRECTION AND PLACES RESULTS IN IC
*     ONLY EVEN VALUES OF N ARE CONSIDERED HERE

      DIMENSION SPIN(128),N(3),IC(3),SNBR(3)
*     CHANGE DIMENSION OF SPIN IF REQUIRED

      IC(1)=IC(2)=IC(3)=0

      DO 1 NROW=1,IMAX
      DO 1 M=1,3
      SNBR(M)=SHIFT(SPIN(NROW),N(M)/2)
    1 IC(M)=IC(M)+IBCOUNT(XOR(SPIN(NROW),SNBR(M)))

      RETURN
      END
```

## TEST RUN OUTPUT

```
LATTICE SIZE: 8 X 8 X 120

ELATT= .6472222222222 EDEM= .007638888888889
ETOT= .6548611111111

AVERAGE OVER 10 ITERATIONS,EACH WITH 10 SWEEPS
RUNNING AT .1299052774019 MFLIPS
BETA= .2224865020859 SISJ= .351177992079
MAGNETIZATION= -.01216927083333

AVERAGE OVER 10 ITERATIONS,EACH WITH 10 SWEEPS
RUNNING AT .1190882307334 MFLIPS
BETA= .2201331059427 SISJ= .3512470296224
MAGNETIZATION= .04859635416667
CORR(IX/4)= .1951041666667
CORR(IX/2)= .1347916666667
CORR(IX/2,IY/2)= .1169791666667

AVERAGE OVER 10 ITERATIONS,EACH WITH 10 SWEEPS
RUNNING AT .1193658688219 MFLIPS
BETA= .2213917811617 SISJ= .351210015191
MAGNETIZATION= .061984375
CORR(IX/4)= .1896875
CORR(IX/2)= .1297916666667
CORR(IX/2,IY/2)= .1181770833333

AVERAGE OVER 10 ITERATIONS,EACH WITH 10 SWEEPS
RUNNING AT .1227425283682 MFLIPS
BETA= .2216398346272 SISJ= .3512027452257
MAGNETIZATION= .001078124999999
CORR(IX/4)= .1961458333333
CORR(IX/2)= .1433854166667
CORR(IX/2,IY/2)= .1174479166667

AVERAGE OVER 10 ITERATIONS,EACH WITH 10 SWEEPS
RUNNING AT .121769462502 MFLIPS
BETA= .2212017631939 SISJ= .3512155897352
MAGNETIZATION= -.1140026041667
CORR(IX/4)= .1984895833333
CORR(IX/2)= .1356770833333
CORR(IX/2,IY/2)= .1197395833333

*** AVERAGES AFTER DISCARDING FIRST BATCH ***
AV. BETA= .2210916212314 +/- .0003318529697922
AV. MAG.= -.0005859374999986 +/- .04000005333508
AV. CORR(IX/4)= .1948567708333 +/- .001862828817412
AV. CORR(IX/2)= .1359114583333 +/- .002808035615709
AV. CORR(IX/2,IY/2)= .1180859375 +/- .0006037980047396
CORR(IX/4)/CORR(IX/2)= 1.43496114271 +/- .022604573658
CORR(IX/2)/CORR(IX/2,IY/2)= 1.151123951271 +/- .02578625876771
```