# A COMPUTER SIMULATION OF COPPER CRYSTAL SURFACE DYNAMICS

Henry William Sterbenz

# NAVAL POSTGRADUATE SCHOOL
Monterey, California

# THESIS

A COMPUTER SIMULATION
OF
COPPER CRYSTAL SURFACE DYNAMICS

Henry William Sterbenz, Jr.

Thesis Advisor: D.E. Harrison, Jr.

June 1972

*Approved for public release; distribution unlimited.*

# A Computer Simulation
# of
# Copper Crystal Surface Dynamics

by

Henry William Sterbenz, Jr.
Captain, United States Army
B.S., United States Military Academy, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
June 1972

# ABSTRACT

A simple physical model of a copper crystal surface was developed. Atoms were considered as quasi-hard spheres which occupied perfect lattice positions. A computer simulation, based on energy consideration only, using the Monte Carlo method was developed, tested and used to study equilibrium surface microstates. As a result of this study, four conclusions were drawn:

1. This model holds promise for further investigation of real crystal surface phenomenon.

2. Minimum energy considerations cause atoms to align themselves preferentially in a ⟨110⟩ direction on the surface of a face-centered cubic crystal.

3. Stepped surface configurations are fairly stable, but isolated "stub" atoms and vacancies tend to coalesce with other "stubs" and vacancies, respectively.

4. Random motions of the individual atoms cause aggregates of atoms to break apart and recombine.

# TABLE OF CONTENTS

## LIST OF TABLES

# LIST OF FIGURES

## ACKNOWLEDGEMENT

There are two people who supported my efforts throughout this project. The first is my advisor, Professor D.E. Harrison, Jr., to whom I am indebted for his many ideas, advice, discussions and friendship. The second is my wife, Ann, without whose understanding, love and assistance I would have never achieved any interesting results. In addition, I would like to thank all the personnel of the W.R. Church Computer Center of the Naval Postgraduate School, for without their daily efforts over several months no computer project of this magnitude could be accomplished. Finally, I extend my thanks to Mrs. Maryann Clarey who so patiently and efficiently typed the manuscript. Of course, all mistakes appearing in the finished product are my own.

# I. INTRODUCTION

## A. HISTORY

This simulation was based on two areas of contemporary research: crystal growth and radiation damage. Ideas were borrowed from each of these fields of study in order to devise a new, simple physical model of a real crystal surface, and to begin the study of crystal surface equilibrium microstates. Brief discussions of prior research is contained in the following paragraphs.

Theoretical models of crystal growth have been studied for over 20 years, and numerous papers have been written attempting to develop models for growth from a solution, a melt, or a vapor [4,6,9,10]. An essential part of these investigations was the assumption of a perfect simple cubic crystal lattice and of first nearest neighbor forces only. Some results have been compared with experimental work done on the growth and dissolution of simple cubic ionic crystals in supersaturated and unsaturated solutions, and have shown both agreement and disagreement with the theory [6]. Motion of atoms absorbed on the surface of the crystal were taken into account by incorporation of theoretical surface diffusion mechanisms based on transport theory. These papers were generally written in terms of classical thermodynamic quantities. Crystal growth, diffusion and dissolution were all discussed in terms of Gibbs free energy and chemical potential for generalization to any simple cubic crystal. The crystals visualized were usually ionic.

Radiation damage studies provided useful physical information on real crystals. Gibson, Goland, Milgram and Vineyard (hereafter, GGMV), and Girifalco and Weizer (hereafter, GW) have investigated interatomic potentials which were found useful. GGMV used a Born-Mayer exponential approximation to the interatomic potential, and applied their results to radiation damage dynamics for short interatomic spacings [5]. The GGMV potential used here has become known as the "Gibson II potential". GW used a Morse potential and fitted data for 15 cubic metals to arrive at a appropriate Morse potential parameters for metals [7]. Anderman followed the GW procedure and obtained slightly different parameters for use in the Morse potential, by introducing the idea of truncating the potential function. Anderman used his values in some static radiation damage simulations [1]. The Anderman potential values have been used here, and the Anderman form of a GW Morse potential will be referred to as a "truncated Morse potential".

Since solving for the motion of the atom in a crystal involves solving an N-body problem, an essential assumption for the radiation damage studies was that the motion of an atom could be described by an independent set of two-body potentials such as the Gibson II and truncated Morse potentials. Such a technique yielded a soluble approxmation to the actual physical problem. For example, an atom and four of its first nearest neighbors gave four independent two-body problems, instead of one five-body problem.

## B. THE NEED FOR NUMERICAL SIMULATION

The detailed processes involved in both of these areas of study were essentially unknown. Both lines of research were attempts to

take macroscopic information and develop microscopic models which predict macroscopic phenomenon. In some cases the theory involved did not permit analytical solution of the problem [6] and in all cases random processes at the atomic level were assumed. Solving the same problem a number of times using a Monte Carlo model and averaging the properties of several final configurations was a technique common to both of these areas of interest. Since the calculations were repetitive and could be programmed fairly simply, computer simulation was a logical tool to use. In addition, since microscopic models were assumed and system microstates were generated, a computer simulation yielded microscopic information - which may be appropriately averaged and interpreted to yield theoretical macroscopic results for comparison with empirical macroscopic results. A "computer experiment" therefore yielded microscopic as well as macroscopic information.

## C. THE MONTE CARLO METHOD

The Monte Carlo method requires that in calculation, whenever one makes a decision based on probability, the decision is made by generating a random number and comparing it with the probability involved. The calculation is continued until it arrives at another decision based on probability. The method then requires the generation of a new random number for comparison and decision. The entire process continues for as long as desired, and a set of final conditions is obtained from a set of given initial conditions. The entire process may then be repeated, starting with the original set of initial conditions and new random numbers, to arrive at a new set of final conditions. In the end there is an ensemble of final

states generated from a single initial state, and it is possible to compute the average properties of the final state [12]. The average final state properties may be compared with experimental data to yield information on the quality of the theoretical microscopic model. Essentially, the method attempts to generate macroscopic information from a microscopic system model.

## D. THE MODEL ATTEMPTED HERE

The present effort is believed to be the first attempt to study the surface dynamics of an actual metal crystal affected only by the temperature of the system. No attempt was made to include growth or dissolution (adding or subtracting atoms) of a crystal. Rather, a simple physical approach was tried in an effort to study only the surface of an actual metallic crystal. What was visualized was a section of a perfect copper crystal surface that was initially "piled" in different ways. Observations were made of the subsequent surface configurations (microstates) resulting from Monte Carlo calculations. Interatomic potentials were borrowed from the radiation damage studies. Transition probabilities were borrowed from the thermodynamic crystal growth studies. An appropriate computer program was written to simulate the copper surface dynamics, and several computer runs were made to obtain a rough idea of what appeared to be equilibrium microstates. The programming language used throughout this study was FORTRAN IV, and the computer used was the IBM 360/67.

# II. THE SIMULATION MODEL

## A. ASSUMPTIONS

Since this was a first attempt at solving a new problem, several simplifying assumptions were made. Perhaps the greatest assumption made in this calculation was that metal crystal binding energies could be computed from a single composite potential energy function, with the interactions between atoms considered as independent two-body problems. The essential feature of this assumption was to treat the atoms of a crystal lattice as quasi-hard spheres which interacted by some "unknown" mechanisms to yield observed crystal behavior.

The only metal visualized was copper, which forms a face-centered cubic (fcc) lattice. For simplicity, the (001) crystal orientation was assumed. Lattice sites were visualized as ordered real triples in the positive octant of a rectangular coordinate system. The coordinate planes and axes bordering the positive octant were considered as belonging to the positive octant. See Figure 1. Although a rectangular parallelepiped was always generated to provide possible lattice locations (hereafter called the "active lattice volume"), appropriate changes in the lattice generator insured that the initial configuration of the surface could take any desired shape with the sole precondition that the only sites available were perfect lattice sites. The active lattice volume was considered as merely a framework of sites that were available for atoms-whether the sites were actually occupied or not depended on the details of the microstate.

In addition, it was assumed that only the top several atomic layers of the surface provide interesting results - in effect, this meant that the active lattice sites generated were placed on top of a perfect lattice substructure which was not allowed to interact with the surface sites in any way other than to be used in calculating surface site potential energy. This further implied that above the active lattice sites all locations were always kept vacant.

Several additional assumptions were made to simplify the model. These were:

1. Atomic vibrations and zero-point energy were neglected.
2. Lattice relaxation due to vacancies was neglected.
3. No interstitial atoms were allowed.
4. No impurity atoms were allowed.
5. Only even numbers of planes in the principal coordinate directions were allowed.

Further discussion of these assumptions is contained in Appendix A.

## B. THE COMPUTER PROGRAM

A computer program that provided an adequate picture of microscopic surface activity had to be written and tested. Overall program flow is shown in Figure 2. The main features of the program are discussed in the remainder of this chapter. Further details are provided in Appendix A.

### 1. The Lattice Generator

The first task involved generating an appropriate latticework of crystal sites in the computer. For the (001) orientation of a face-centered cubic crystal (hereafter, "fcc, (001) crystal"),

available sites are described by coordinates such that the sum of coordinates is either an even or an odd integer for all positions in the crystal. The even choice was used in this simulation.

After a valid lattice site was located, it was given a number as a label. Then an occupation index was defined for the site to retain a knowledge of the surface configuration. Several different initial configurations were used, and these are sketched as the "a" parts of Figures 3 through 17. The specific program statements to generate such surfaces are contained in Appendix B.

2. Periodic Boundary Conditions

Computer storage and time limitations made impractical the use of an arbitrarily large active lattice volume. A practical lattice size involved at most a few thousand active lattice sites. This required that the active volume size be limited to a Cartesian space of roughly 20 x 20 x 10, which corresponds to 2,000 active lattice sites and a copper crystal measuring about 50Å x 50Å x 25Å. Such a small crystal with six surfaces complicated the study of surface effects.

One surface (the bottom of the active volume) has already been eliminated by assuming that the active lattice volume was superimposed on a perfect crystal substrate which was never allowed to vary. Four other surfaces (the sides of the active volume) were eliminated by assuming "periodic boundary conditions" which resulted in an essentially infinite single surface (the "top" of the active volume) that varied periodically in configuration. This infinite surface may be visualized as part of a crystalline surface far from a grain boundary.

These periodic boundary conditions were achieved by visualizing that each active volume plane was laterally surrounded by exactly similar active volume planes that had been translated in the x and/or y direction(s). In other words, an atom on the left edge of an active volume plane "sees" the right edge of that same active volume plane; an atom on the top edge "sees" the bottom edge of the same plane, and similarly for the other edges.

Periodic boundary conditions were achieved in practice by defining nearest neighbor (NN) arrays for each site. These arrays were calculated by visualizing the spatial relationships between an atom in the fcc, (001) crystal and its nearby surrounding sites - see Figures 18 and 19. Atoms within one atomic layer of the surface of the active lattice volume required special attention to insure the proper periodic structure of each plane.

### 3. Potential Energy

The model chosen assumed that the crystal may be approximated by a latticework of quasi-hard spheres, so a scheme for holding individual atoms together was needed. The problem was an N-body problem, and therefore no general exact solution existed. However, approximating the potentials by an independent set of two-body potentials yielded a solvable approximation to the physical system, and this was a standard technique used in radiation damage studies.

The next task involved a judicious choice of potential function. A composite potential function was chosen. For low values of interatomic spacing the Gibson II potential [5] was used, and for larger values the truncated Morse potential [1] was used.

Since these two potentials do not match smoothly together (see Figure 20), a cubic function was generated which matched the value and the slope of these two potentials at appropriate points so that a smooth composite potential was achieved (see Figure 21). The cubic potential coefficients were generated by solving a system of four simultaneous linear equations using Subroutine CROSYM developed at Lockheed. The form of the potential equations, their appropriate coefficients, and the cutoff values used in this simulation are listed in Table I.

Next, a potential energy was associated with each site in the lattice based on the number of its first and second nearest neighbor (NN1 and NN2, respectively) sites that were occupied. In other words, every active volume lattice site was given a potential energy. If the site were occupied, this energy was just the total mutual interaction energy between an atom and its NN1's and NN2's. However, if the site were vacant, this energy was the interaction energy between the NN1's and NN2's of the site that would arise _if_ the site were occupied. In practice, potentials were given to each site (occupied or vacant) in the same manner, but the use to which these potentials were put depended vitally on whether a particular site was occupied or not.

On the average about half of the mutual interaction energy between sites will be associated with each site individually, so a parameter PFIV (mnemonic for "point five", and usually equal to 0.5) was used as an energy distribution factor. By varying this energy distribution factor, physical effects resulted which were equivalent to varying the lattice temperature. Therefore, by

changing the energy distribution factor, different lattice temperatures were achieved. The relationship between the equivalent lattice temperature and PFIV is indicated in Figure 22 and discussed in greater detail in Appendix A.

4. Transition Probability

If a given lattice site were occupied and had one or more vacant sites as NN1's it had (statistically) a finite probability of moving from its present site to the nearby vacant site. In practice, when an occupied lattice site was reached, the computer searched for lower values of site potential among the unoccupied NN1's of the occupied site of interest. Such lower energy vacant sites implied that the atom under consideration was sitting on the edge of a potential well, and a transition was carried out to the deepest NN1 well - that is, the atom "jumped" into the deepest NN1 well. If a "deepest well" was not unique, then the actual final site was chosen stochastically. "Wells" of zero depth were considered as having "lower energy" for the purpose of this calculation.

For atoms with no vacant NN1 sites of lower energy but one or more with higher energy, the atom would have to climb a potential hill to reach it (them). This was considered possible but was given a probability defined by a Boltzmann factor:

$$p = \exp\ (-E/kT) \qquad (1)$$

where

P = the probability of transition to a higher energy site

E(>0) = energy difference between the occupied site of interest and one of its NN1 sites with higher energy

k = Boltzmann's constant

T = absolute temperature

The computer searched for the lowest positive energy difference and made the decision to jump or not stochastically. If several NN1 sites had positive energy differences, they were considered in turn from the lowest to the highest until a transition was made. If no transition were made by the time the positive energy differences were exhausted, then the atom stayed where it was for that particular microstate.

After any transition the potential energy of the NN1's and NN2's of both the old and new sites were adjusted to account for the changes made. After every third microstate, all potentials were recalculated to avoid loss of accuracy. A pseudo-random number generator was used to produce the random probabilities needed to make the decisions involved.

### 5. Computer Output

The problem of how to present microstate information in easily interpreted form was solved by having the computer "draw" pictures of microcrystal planes, called "microstate pictures". The computer printed arrays of occupation indices to indicate a vacancy or an occupied site, and each digit was in its correct x and y coordinate position for the site it represented. See Figure 23.

# III. RESULTS

## A. INITIAL TESTING

Test runs made prior to taking any "good data" indicated that two problems would be encountered. The first problem discovered was the lack of net motion out of a perfect surface configuration. The second problem was an indication that the model might tend to "transport" atoms preferentially in any given plane from higher to lower values of the arbitrarily assigned y-coordinate.

The problem of no net motion from a perfect surface was solved by two different techniques. One was to alter the energy distribution factor PFIV to define an equivalent lattice temperature to literally force some action. The other method involved piling the surface atoms in both random and preconceived configurations for the initial microstate and observing subsequent microstates using the correct average value of PFIV (= 0.5). The results of these tests are discussed in Sections B and C, below.

The problem of model dependent transport was more fundamental, since any results obtained would be biased in some unknown fashion. Again, two techniques were used to explore the problem. The first technique was to test the motion of a single atom initially placed on a perfect surface. The motion of this single "stub" atom was observed for 400 microstates. The motion within each microstate was recorded as the net number of lattice units moved "up", "down", "left" or "right" from its previous position. These directions were defined for an observer looking at a single lattice plane;

"right" meant "in the + x-direction", and "up" meant "in the + y-direction" for the plane under consideration. The average results are reported in Table II to the nearest one percent. A quick glance at the figures appeared to lead to the conclusion that the model was biased to motion right and down. However, consideration of the average number of jumps weighted by the distance jumped indicated that:

1. When the stub moved left, its average motion was 0.47 lattice units.

2. When the stub moved right, its average motion was 0.61 lattice units.

3. When the stub moved up, its average motion was 0.46 lattice units.

4. When the stub moved down, its average motion was 0.44 lattice units.

This meant that the average net motion of a single stub atom was right and almost entirely lateral.

Since this test was performed with only one stub atom, a second technique was used to test for model biasing. The technique was to study the apparent motion, if any, of large numbers of atoms placed on a perfect surface. This required, in effect, "taking data" and the results were available only after several long computer runs were made. Based on the runs discussed below, it was concluded that there existed no significant motion of large numbers of atoms attributable to model biasing. Some motion "down" was found, but this amounted to about one lattice unit every 10 to 20 microstates for 100 or more atoms and was not considered

significant. This observation was purely qualitative, and such motion was not found in every case. In addition, no gross lateral motion was observed, and any that did exist was much slower than the "down" motion.

## B. PERFECT SURFACES

Tests made using perfect surfaces required that the energy distribution factor PFIV be altered. The arbitrary initial temperature chosen for all calculations was 1,000 °K. With the correct average value of PFIV (= 0.5), no net change was observed for any microstate in 500 attempted microstates when the initial microstate was a perfect surface.

In an effort to force some different microstates to occur, PFIV was lowered by a factor of 10, yielding an equivalent temperature of 10,000 °K - well above the copper boiling point of 2,583 °K. This resulted in a decidedly unphysical sequence of surface microstates. Since the temperature was so high, the surface literally evaporated and burroughed a deep hole into itself within 10 microstates. The model restraints insured that no atom would be lost, but this did not mean that the microcrystal did not try to "explode" - a program flag included to warn that an atom had no NN1's was flashed at least 35 times during simulation of 500 microstates and never appeared in any other simulation. Equilibrium numbers of atoms in each plane was reached after about 30 microstates (as opposed to more than 100 microstates normally required for other simulations). Some single atom vacancies and stubs were observed, but such surface defects were repaired quickly. However, the frequency of observation of these defects was much greater in

this simulation than for all other simulations. Figure 24 shows a typical microstate achieved. Note that the figure is drawn with lines bordering the atoms and atom clusters in a ⟨100⟩ direction (Figure 24a) and in a ⟨110⟩ direction (Figure 24b). For all other simulations, lines parallel to a ⟨110⟩ direction appeared to be most natural to a face-centered cubic crystal, but here the jagged edges appearing in both sketches indicated that neither was "natural". Chapter IV contains a discussion of the "natural" crystal orientation. Also note that there are several "overhangs" visible in each sketch - overhangs were observed in other simulations, but not with the great frequency with which they appeared here.

The next perfect surface attempted was at an equivalent temperature of 1,355 °K - just one degree Kelvin below the copper melting point. In 200 microstates no net change was observed for any microstate.

Next, a perfect surface at an equivalent temperature of 2,000 °K was attempted. A typical microstate achieved is indicated in Figure 25. The atoms making up the top perfect surface layer "jumped up" and coalesced on the surface.

Finally, a simulation at 2,584 °K (one degree Kelvin above the copper boiling point) was attempted. Equilibrium numbers of atoms in each plane were reached by about microstate 100. A typical microstate is indicated in Figure 26. Atoms in several surface layers jumped up and actually formed steps with edges parallel to the ⟨110⟩ direction.

## C. PERFECT SURFACES PLUS EXTRA ATOMS

Most simulations attempted fell into this category. This type of initial configuration was used for two reasons:

1. It was unphysical to expect that real crystal surfaces would be absolutely perfect for large sections of surface.

2. It was unphysical to require temperatures greater than the melting point in order to achieve some idea of what surface equilibrium microstates were.

All of these simulations were run at a lattice temperature of 1,000 $^{o}$K and using the correct average value of the energy distribution, PFIV (= 0.5).

Two simulations were run placing atoms randomly on top of a perfect surface. Different random number seeds were used for each simulation, resulting in different numbers of extra atoms on top of a perfect crystal surface. See Figures 3a and 4a. The randomly placed atoms coalesced within 20 microstates in both simulations. Figures 3b and 4b show typical equilibrium configurations for each experiment. Note that the atoms arranged themselves preferentially along the $\langle 110 \rangle$ direction.

One simulation was run placing an extra half plane of atoms on top of a perfect surface, with the halfplane edge parallel to the x-axis. See Figure 5a. At equilibrium, the atoms were still together but again showed a preference for lining up parallel to the $\langle 110 \rangle$ direction. See Figure 5b.

Two simulations were run piling atoms into a pyramid configuration with sides of different slopes. Sketches of the pyramid initial microstates may be found as Figures 6a and 7a with the

atoms aligned in the ⟨100⟩ and ⟨010⟩ directions. The final configurations are shown in Figures 6b and 7b. Note that in Figure 6b all piled up atoms have fallen down to coalesce on top of the original base plane. Note that in Figure 7b not all of the atoms have fallen down - in fact, there are aggregates of atoms in four planes above the original base plane. The large "hole" appearing in the plane above the base plane is the outer "groove" appearing in the original microstate that has been reoriented and partially filled in. Note that one atom has actually jumped out of the base plane and has not been replaced - this is represented by the "hole" in the left corner of Figure 7b.

Two simulations were run with the atoms arranged to form a "pyramid hole" of different step widths - see Figures 8a and 9a. The configurations 500 microstates later are shown in Figures 8b and 9b. Note that in both cases the atoms have fallen down into lower planes, but that the "falling down" process is a bit slower when the hole step is 2 atomic layers wide then it is for a hole step of 3 atomic layers. The reason for this is probably the fact that with 2 atomic layers per step there are more atoms in the active lattice volume, and the active volume itself is larger.

Three simulations were run piling atoms into ridges - one was parallel to the x-axis and the other two were parallel to the y-axis. See Figures 10a and 11a. Despite the different ways the atoms were initially piled and despite the fact that different random number seeds were used for the two ridges paralleling the y-axis, there was a remarkable resemblance in the gross appearance of equilibrium microstates of each experiment - these are sketched

as Figures 10b, 11b and 11c. The slots appearing in these sketches are all in the x = 1 plane, which was not completely filled in the initial microstates, i.e., the holes do not represent several atoms jumping out of the z = 1 plane but the rearrangement of a hole originally placed there after it has received some additional atoms from the higher planes.

Next, four simulations were run in which the atoms were piled into valleys of different step widths. The initial microstates are sketched in the "a" parts of Figures 12 through 15. Note the similarity between Figures 10a and 12a and between Figures 11a and 14a - in effect, since periodic boundary conditions were applied, Figure 12a may be obtained from Figure 10a by translating Figure 10a one-half the active volume distance in the y-direction, and similarly for Figures 11a and 14a with translation in the x-direction. In any event, it may be expected that some overall gross similarity of the final microstates may appear. Comparison of Figures 10b and 12b, and Figures 11b and 14b tends to verify this expectation. Note that again the holes appearing in the final microstates were a result of rearrangements and partial fill-in of the original holes in the z = 1 planes. Figures 13a and 15a indicate valleys of narrower step width, and final microstates achieved are sketched as Figures 13b and 15b.

Since there was a preference for alignment along the ⟨110⟩ direction, the two final simulations started with all atoms aligned in that direction. One was visualized as a truncated (1 -1 1) plane (Figure 16a), and the other as a truncated (-1 1 1) plane (Figure 17a). The 500th microstates are sketched as Figures 16b

and 17b. Note that in each case atoms were still piled high and no plane was completely filled - these facts indicate that the system probably had not reached equilibrium. These initial microstates were a bit more unphysical than the others tested since they started with steps several atomic layers high on 2 sides and one atomic layer high on the other. This permitted overhangs when atoms "jumped off" the steep sides, but this was not considered seriously detrimental to the results.

# IV. CONCLUSIONS AND RECOMMENDATIONS

## A. CONCLUSIONS

As indicated in the discussion of the perfect surface simulation results, most "experiments" yielded results which imply that the (110) orientation was most natural for sketching equilibrium microstates. Here, "natural" meant that there were fewer jagged edges and protruding atoms than if the microstate were sketched in the (100) orientation. Note that the initial configuration sketches (the "a" part of Figures 3 through 17) are all in the (100) orientation, since this was how the crystal was initially visualized and developed. The following discussion will neglect the one unphysical case achieved - that of an initially perfect surface at 10,000 $^{o}$K.

The first important result of testing this model was that there was one thread of similarity running through all the experiments - in the final microstates achieved, all atoms coalesced into steps either parallel or perpendicular to a ⟨110⟩ direction. This was most encouraging since the initial state had no effect on the final step direction. It was further encouraging since when building an fcc, (001) crystal from marbles and glue, atoms on the surface are more closely arranged in a ⟨110⟩ direction than in a ⟨100⟩ direction. The physical reason for such a preference lies in energy considerations: in an fcc, (001) crystal, NN1's are located along ⟨110⟩ directions, and NN2's along ⟨100⟩ directions. Therefore, the lower energy state sought by a system intuitively and deliberately

built into the transition probabilities of this model indicate a preference for the ⟨110⟩ direction. Again, it is a pure energy consideration which dictated that the atoms coalesce and not disperse randomly. The minimum energy condition was achieved for as many atoms as possible staying close together. Occasionally, isolated surface vacancies or stub atoms would appear, but they were quickly repaired or moved so that these defects tended to coalesce.

The second conclusion involved the final shape of piled atoms. Atom piles tended to be stable in a stepped configuration over several microstates, and this agreed with preconceived notions of surface shape. Figure 27 shows a sketch due to D.E. Harrison [8] indicating his concept of a surface equilibrium microstate. According to the model developed and tested here, the sketch is essentially correct if it is drawn in a (110) orientation for its stepped features. However, this model indicated that isolated holes and stubs do not persist at equilibrium. Isolated vacancies or "holes" were as mobile as isolated stub atoms. Isolated groups or atoms and holes appeared to move slowly as a group to coalesce with larger aggregates.

The third observation of importance involved breaks and their repair in elongated strings of surface atoms. Figure 28 shows a series of sketches illustrating this phenomenon. When the motion of the atoms led to the movement of atoms in opposite directions, a local "necking down" of surface aggregates of atoms was observed. Random motions led to selective breaks at such thin steps, but unless the atoms in the two resulting steps moved several atomic diameters apart the break actually repaired itself. Conversely,

unless the break repaired itself by supplying several atomic diameters of step, the break was likely to recur within 10 or 20 microstates. This behavior was observed repeatedly, and was not really surprising although it was not expected beforehand.

## B. RECOMMENDATIONS

First, it is recommended that this model be further studied and refined to obtain a more realistic picture of metal surface microstates. The results achieved and conclusions drawn here must be considered only as preliminary indications since this was a first attempt at a new physical model and since the testing was not extensive.

Secondly, it is recommended that prior to extensive experimentation to find equilibrium microstates, the problem of model dependent transport be more thoroughly investigated. Since large aggregates of atoms provided the most interesting results here, this testing should be carried out with groups of at least 10 atoms on a perfect surface. It is envisioned that a practical test might involve calculating the center of mass of these aggregates for several hundred microstates and computing the average motion of their center of mass. If trends appear (as in the case of a single stub atom, tested here), then appropriate corrections must be made in the computer program. Perhaps the easiest possible change to install involves the sequence in which the atoms are looked at as a microstate is generated. As it was used here, every microstate was generated by having the computer investigate each atom in order by lattice site number, which meant that rows

were investigated from left to right and from bottom to top in each z-plane. It may be more realistic to generate every other microstate in a different way, e.g., by investigating each column from bottom to top and from left to right in each z-plane.

Third, it is recommended that the following changes be made in order to improve the model:

a. Rewrite this fcc program to generate the active lattice volume in a (110) orientation. This would require a new lattice generator, new lattice loading steps, new NN assignment segments, and recalculation of potential energies.

b. Include consideration of NN3's and NN4's - this would require new input parameters for the truncated Morse potential plus NN3 and NN4 assignment segments.

c. Allow floating point values for the atom positions and introduce kinetic energy for the atoms. It is expected that introducing a normal distribution of kinetic energies would reduce the total energy of perfect surface atoms sufficiently to start them "jumping" out of their positions without resorting to the equivalent temperature idea introduced here. This would allow for the introduction of lattice temperature as an input parameter and controlling factor in lattice activity.

d. Write a lattice loader that would generate a more realistic lattice - e.g., one in which the atoms were randomly placed subject to physical restraints such as having at least one NN1 and occupying nearly perfect lattice positions (i.e., this assumes that floating point lattice positions have already been incorporated).

e. Allow for vacancies, interstitials and impurities. This would require consideration of lattice relaxation and strain effects, and would require new potentials for impurity-copper and impurity-impurity interactions.

# APPENDIX A

## THE DETAILS OF THE SIMULATION MODEL

### A. ASSUMPTIONS

Perhaps the greatest assumption made in this calculation is that metal crystal binding energies can be computed from a single composite potential energy function, with the interactions between atoms considered as independent two-body problems. The essential feature of this assumption was to treat the atoms of a crystal lattice as quasi-hard spheres which interact by some "unknown" mechanisms to yield observed crystal behavior. The "unknown" mechanisms include, in the general case, attraction due to ionic, covalent, metallic and van der Waals forces, and repulsion due primarily to electron cloud overlap. The specific mechanisms, however, were of no concern. No quantum mechanical considerations were attempted - that is, the calculations involved were all classical once the appropriate composite potential function was assumed. First and second nearest neighbor interactions were considered.

The only metal visualized was copper, which forms a face-centered cubic (fcc) lattice. For simplicity, the (001) crystal orientation was assumed. Lattice sites were visualized as ordered real triples in the positive octant of a rectangular coordinate system. The coordinate planes and axes bordering the positive octant were considered as belonging to the positive octant. See Figure 1. Although a rectangular parallelepiped was always generated to provide possible lattice locations (hereafter called the

"active lattice volume"), appropriate changes in the lattice generator insured that the initial configuration of the surface could take any desired shape with the sole precondition that the only sites available were perfect lattice sites. The active lattice volume was considered as merely a framework of sites that were available for atoms - whether the sites were actually occupied or not depended on the details of the microstate.

It was further assumed that atomic vibrations about lattice sites could be neglected, so there were no zero-point energy considerations or fluctuations in lattice potential due to atomic motion. Lattice relaxation due to crystal vacancies was ignored, as were interstitial atoms and their consequent lattice strain effects. Furthermore, no impurity atoms were allowed. With these assumptions, it was possible to describe each available lattice site as ordered integer triples and use a single composite potential energy function for every two-body pair in the system. This simplified the model considerably and yielded a great reduction in the potential energy calculations required.

In addition, it was assumed that only the top several atomic layers of the surface would provide interesting results - in effect, this meant that the active lattice sites generated were placed on top of a perfect lattice substructure which was not allowed to interact with the surface sites in any way other than to be used in calculating surface site potential energy; this further implied that above the active lattice sites all locations were always kept vacant. Specifically, lattice sites described by $z<0$ were considered always full - see Figure 1. Lattice sites

described by $0 \leq z \leq z_{max}$ were considered as available for containing an atom, depending on the particular microstate initialized or achieved during a calculation. Lattice sites described by $z > z_{max}$ were considered always empty.

Finally, this model was restricted to even numbers of planes in each of the principal coordinate directions to simplify the calculations involved in assigning first and second nearest neighbors to each available site. This means that $x_{max}$, $y_{max}$, and $z_{max}$ must all be odd, since each coordinate plane was included as part of the active lattice.

## B. THE COMPUTER PROGRAM

### 1. The Lattice Generator

a). Logic. Before any calculations could be made, the computer had to be told what the available sites for atoms were and whether or not these sites had an atom - that is, whether a given site was occupied or vacant. For the (001) orientation of an fcc crystal (hereafter, "fcc, (001) crystal"), the essential technique involved relied on the fact that the sum of the site coordinates must be an even (or an odd) integer (this point is particular to this lattice shape and orientation). Refer to Figure 29. For example, the Cartesian point (1, 1, 1) has a sum of coordinates equal to three, an odd integer - therefore, this point is not an acceptable lattice location. On the other hand, Cartesian points (1, 0, 1) and (1, 1, 0) each have a sum of coordinates equal to two, an even integer - these points are acceptable lattice locations and are assigned integer labels when they are reached in the lattice

generator segment of the program. Figure 30 shows the numbering scheme for lattice points.

The lattice was generated by a set of nested DO loops. One plane at a time was generated, and these planes were parallel to the x-y plane. The coordinates of each acceptable site and its occupation index were stored as vector arrays. An "occupation index" (vector "NOCC(I)") merely indicated whether or not a site was occupied: an index of zero implied that the site was vacant, while an index of one meant that it was occupied. No values other than zero or one were allowed for occupation indices.

b). The Initial Microstate. In order to provide adequate testing of the simulation model, the initial microstate must be easily changed. This was achieved by substituting program segments containing one or more cards into the program. Initial microstates actually used were:

1) Perfect surface

2) Perfect surface plus extra atoms

   (a) randomly dispersed extra atoms - Figures 3a and 4a

   (b) a monatomic step - Figure 5a

   (c) pyramids - Figures 6a and 7a

   (d) pyramid holes - Figures 8a and 9a

   (e) ridges - Figures 10a and 11a

   (f) valleys - Figures 12a, 13a, 14a and 15a

   (g) (1 -1 1) plane - Figure 16a

   (h) (-1 1 1 ) plane - Figure 17a

The actual program segments which generate these different initial conditions are listed in Appendix B.

A site labeled "IFUL" was used to represent all sites in the crystal substrate, that is, all sites for $z < 0$; since these sites were always full (occupied), NOCC(IFUL) was always one. A site labeled "IVAC" was used to represent all sites above the active lattice volume, that is, for $z > z_{max}$; since these sites were always vacant, NOCC(IVAC) was always zero. These assignments were made immediately after the lattice generator and were based on the last value of "M" reached in the lattice generator.

2. Periodic Boundary Conditions

a). The need. Computer storage and time limitations made impractical the use of an arbitrarily large active lattice volume. A practical lattice size involved at most a few thousand active lattice sites. This required that the active volume size be limited to a Cartesian space of roughly 20 x 20 x 10, which corresponds to 2,000 active lattice sites and a copper crystal measuring about 50 Å x 50 Å x 25 Å. Such a small crystal with six surfaces complicated the study of surface effects.

b). The method. One surface (the bottom of the active lattice volume) was eliminated by assuming that the active lattice volume was superimposed on a perfect crystal substrate which was never allowed to vary. Four other surfaces (the sides of the active volume) were eliminated by assuming periodic boundary conditions which resulted in an essentially infinite single surface (the "top" of the active volume) that varied periodically in configuration. This infinite surface may be visualized as part of a crystalline surface far from a grain boundary.

The overall scheme was represented as follows (see Figure 31): Each active volume plane was laterally surrounded by exactly similar active volume planes that had been translated in the x and/or y direction(s). In Figure 31a, the actual active volume was visualized as the center square; the remaining eight squares were the "boundary volume planes" which had exactly the same configuration as the center square. Square 2 was achieved by translating square 1 a full microcrystal distance in the "-x" direction, and square 3 was achieved by translating square 1 a full microcrystal distance in the "+x" direction. Squares 4-5-6 were achieved by the similar process of translating squares 2-1-3 a full microcrystal distance in the "+y" direction, and similarly for squares 7-8-9.

The active volume z-plane of Figure 31b is portrayed as having six x-planes and six y-planes, and a total of 18 lattice sites; three of these lattice sites are labeled (A,B,C), and the location of the remaining sites are indicated by X's. The images of the labeled sites are indicated in each of the surrounding boundary planes. Without the boundary planes, a vector [1,1,0] from active volume site "C" would yield a point in free space. However, with the boundary planes, as in Figure 31b, that same vector from "C" yields a site labeled "B" in the boundary plane. In effect, looking to the right from the active volume plane, one passes through infinity and comes upon the left edge of the same active volume plane in the same row from which one started. Similarly, a vector [0,6,0] from site "A" brings one right back to site "A". (This is merely an illustration, and actual crystals

generated typically had 20 or so planes in the x and y directions, so that returning to the same site was not so easy in practice.) During the calculation, any image sites reached were treated as active volume sites. Specifically, if in the active volume, sites A and C were occupied, but site B was vacant, then in all eight boundary planes, sites A and C were occupied and site B was vacant. This gave a periodic character for site occupation as one moved across a given z-plane, hence the name periodic boundary conditions. In actuality only one active volume was used, but the "image site" and "boundary plane" ideas explained here help to illustrate how periodicity was achieved.

c). Second nearest neighbor assignment segment. The NN2 assignment segment is easier to understand than the NN1 assignment segment, and so is considered first. Figure 19 illustrates the positions of the six NN2's in a fcc, (001) crystal, using the numbering scheme developed for the program. To find the site number of any second nearest neighbor of a given atom, the given atom was visualized as the center of coordinates of a small mobile coordinate system as depicted in Figure 19. In general, the site at the origin was labeled "I", and the six positions for NN2's were labeled "J", where J varied from one to six. A double subscript notation was chosen and named array "NBRTWO(J,I)" - a mnemonic for "neighbor two" - where subscript J refers to one of the six NN2 positions, subscript I refers to the label of the lattice site being investigated (given to that site by the lattice generator), and the value stored at the particular position was the label of the lattice site at position J with respect to site I.

In the general case, NN2 position one was labeled site number (I-1), and NN2 position two was labeled site number (I+1). NN2 positions three and four required that a shift of two rows of atoms in the "-y" and "+y" directions be made. For even numbers of x planes, that is, for IX even, the appropriate number of site numbers to "shift" is IX, since each row of sites in a given plane contains IX/2 sites (IX even). Consequently, NN2 position three was labeled site number (I-IX) and NN2 position four was labeled site number (I+IX). NN2 positions five and six required a shift of two planes of atoms in the "-z" and "+z" directions. Since each row of a given plane contained IX/2 sites (IX even), and there were IY such rows per plane, each plane contained (IX/2)*(IY) sites. Therefore, two such planes contain IX*IY sites. As a result, NN2 position five was labeled site number (I-IX*IY) and NN2 position six was labeled site number (I+IX*IY).

For sites on, or one atomic layer in from, the active volume surface, proper labeling of NN2 sites helped achieve the desired effect for a single infinite surface varying periodically. For example, a site with NX(I)=0 (on the left face of the active volume) or NX(I)=1 (one atomic layer in from the left face of the active volume) looking for its neighbor at NN2 position one must "see" that neighbor (call it I') with NX(I')=$z_{max}$-1 or NX(I')=$z_{max}$, and NY(I')=NY(I), and NZ(I')=NZ(I). In the site labeling scheme chosen, this represents a shift of (IX/2)-1 site numbers (IX even). Similar shifts must be made when "looking through infinity" from the other three sides of the active lattice volume, and this was

essentially how the imposed periodic boundary conditions were achieved in practice.

For sites with NZ(I)=0 or NZ(I)=1 looking for their neighbor at NN2 position five, site IFUL (the always full substrate position,was reached. Similarly, for sites with NZ(I)=$z_{max}$ or NZ(I)=$z_{max}$-1 looking for their neighbor at NN2 position six, site IVAC (the always empty above lattice position) was reached. In this way a perfect substrate below the active volume and a void above the active lattice were maintained. Further specific details of the NN2 assignment segment are omitted here, but may be found by studying the actual program between statements number 195 and 201.

d). First nearest neighbor assignment segment. The NN1 assignment segment was approached in much the same way as the NN2 assignment segment. The problem was complicated by the fact that there are 12 NN1's for an fcc, (001) crystal, and there are 22 special cases (6 faces, 12 edges, and 4 corners) to consider, as opposed to six NN2's and six special cases for the NN2 assignment segment. Figure 18 illustrates the positions of the 12 NN1's, using the numbering scheme developed for the program. Again, a double subscripted array was chosen to store NN1 information, and it was called "NBRONE(J,I)" - a mnemonic for "neighbor one" - where subscript J refers to one of the 12 NN1 positions, subscript I refers to the lattice site under consideration, and the value at the particular array position is the site number of the site at position J with respect to site I.

The initial complication that arose was an "even-odd" dependence on the x-value of the site for 8 out of 12 NN1 positions. Taking this into account, the overall problem was solved by first assigning each site its 12 NN1's without regard to the 22 special cases. These were computed using a shift technique similar to that developed in the NN2 assignment segment. The 22 special cases involved sites on the faces, edges, and corners of the active lattice volume, and required particular attention in order to appropriately achieve the desired periodic boundary conditions. The method used then involved testing for each of the 22 special cases. As each special case was found a series of from four to nine adjustments were made from the "general NN1 sites" already stored. Sites IFUL and IVAC were employed in a manner similar to that involved in the NN2 assignment segment. Further details of the calculation are tedious and will not be examined here. The actual procedure may be gleaned from the program itself, beginning after the lattice generator and running through statement 195.

Thus, the periodic boundary conditions imposed on the active lattice volume were achieved in practice via the use of the NN1 and NN2 assignment segments. These segments, as written, required that IX, IY and IZ all be even. This was not considered a very severe restriction, and there is no a prior reason to expect that the results were biased due to a choice of even numbers of planes in the coordinate directions.

### 3. Potential Energy

a). The Composite Potential Energy Function. The model chosen assumed that the crystal may be approximated by a

latticework of quasi-hard spheres, so a scheme for holding individual atoms together was needed. The problem is essentially an N-body problem, and therefore no general exact solution exists. However, approximating the potentials by an independent set of two-body potentials yields a solvable approximation to the physical system, and was a standard technique used in radiation damage studies.

Since the approach taken involved the solution of independent two-body potentials, the next task involved a judicious choice of potential function. The choice began with the Morse Potential, proposed in 1929 as an appropriate potential for the wave equation of a diatomic molecule [13]:

$$\text{(A-1)} \qquad V_{\text{Morse}} = D \exp(-2a(r-r_o)) - 2D \exp(-a(r-r_o)),$$

where

$D$ = dissociation energy of the molecular bond

$r_o$ = equilibrium interatomic separation

$r$ = interatomic separation

$a$ = constant related to the curvature about the potential minimum of the function

Note that the first term dominates for small "r", and that the second term dominates for large "r".

Girifalco and Weizer calculated Morse potential constants for 15 cubic metals, and these have found satisfactory agreement with experiment [7]. Anderman used essentially the same procedure as Girifalco and Weizer, but used a truncated potential and obtained parameters for a copper Morse-type potential for two, three, and four nearest neighbors [1]. This "truncated Morse"

potential with parameters for two nearest neighbors was used for the range $r_B < r < r_C$, where $r_B$ is an inner cutoff value chosen as a lower bound for the truncated Morse potential, and $r_C$ is the truncation distance.

GGMV arrived at reasonable short range results using a Born-Mayer type exponential approximation for the interatomic potential:

$$\text{(A-2)} \quad V_{\text{Gibson II}} = D \exp(-a(r-r_o)),$$

where D, a, r and $r_o$ have the same meanings as in $V_{\text{Morse}}$. This "Gibson II" potential was found to be a good approximation for small interatomic separations, and so was used for short distances, $0 < r < r_A$, where $r_A$ is an outer cutoff value chosen as an upper limit for the region of Gibson II potential validity. In general, $r_A \neq r_B$ because these two potentials do not match smoothly - see Figure 20. The "mismatch region" was fitted by a series expansion with a cubic function, since only four pieces of information were available to match as boundary conditions (these were the value and slope of the Gibson II potential at $r_A$, and the value and slope of the truncated Morse potential at $r_B$) - the most general cubic

$$\text{(A-3)} \quad Ar^3 + Br^2 + Cr + D$$

has four unknowns, which just equals the information available. The cubic is a standard choice for this "linking potential".

Solving for the coefficients of the cubic potential involved solving four simultaneous linear equations in four unknowns. The technique of solution involved here required putting the coefficients of the four linear equations into a matrix which

was then transferred to Subroutine CROSYM (developed at Lockheed) which performed manipulations on the matrix (using the "Method of Crout"). An answer vector was coupled back to the MAIN program.

The actual potential function used in the program was a composite potential, employing:

the Gibson II potential, for $0 < r < r_A$,

the cubic potential, for $r_A \le r \le r_B$,

and the truncated Morse potential, for $r_B < r < r_C$.

Figure 21 shows the overall composite potential. Note that all interatomic separations used in this program were greater than $r_B$, that is, the cubic and Gibson II potentials were never used, in practice. However, they were included to enchance program flexibility, since it was anticipated that future improvements of this model would allow atomic vibrations which may require the use of these other potentials. All these formulas are for interatomic spacings in lattice units. A lattice unit is the distance between two NN1's in a single coordinate direction.

b). Site Potential Energy. The next step was to assume that each lattice site experienced this composite potential as the appropriate two-body interatomic interaction potential. Program statements 1050 through 1100 set the square of the equilibrium interatomic separation with the simple formula

(A-4) NNDIS2(N) = 2*N.

This formula is exact for the fcc, (001) crystal for NN1 through NN13 in lattice units. Using these values for distance squared, a potential was computed for each ith nearest neighbor (i=1,2) based on its distance from any particular site. Consider the sketch in

Figure 21. The program computed and stored $V(r_1)$ and $V(r_2)$. It then looked at each site and (whether it was occupied or not), it computed an energy associated with that site based on nearby occupied sites out to and including the second nearest neighbor sites. This energy was then multiplied by PFIV (a mnemonic for "point five" and equal to 0.5 typically, but included as a convenient parameter for calculation) since on the average about half the potential was associated with each atom of the many two-body pairs envisioned. Values of PFIV other than 0.5 gave a different value of effective lattice temperature, and in practice lattice temperature was varied by changing PFIV. The true equivalent lattice temperature based on potential energy only was defined by

(A-5) EQTEMP = 0.5*TEMP/PFIV,

where EQTEMP is the equivalent lattice temperature

TEMP is the input temperature used (=1000.0 $^{o}$K).

Figure 22 is a log-log plot of (A-5).

In this way each site was given a potential energy based on nearby occupied sites, These energies were stored as the vector PE(I), where I was the site number. An energy associated with both occupied and vacant sites was required in order to determine relative transition probabilities from occupied lattice sites to nearby vacant sites - and this transition probability was related to the energy change between sites by what was essentially a Boltzmann factor.

### 4. Transition Probability

a). Definition. If a given lattice site was occupied and had one or more vacant sites as NN1's, it had (statistically) a

finite probability of moving from its present site to the nearby vacant site. In practice, when an occupied lattice site was reached, an energy difference array was defined between the site reached and each of its 12 NN1's. The array was defined as "DE(J,I)" - a mnemonic for "difference in energy" - where I was the occupied site reached and J (running from 1 through 12) referred to the position of one of the NN1's for site I. The value stored at the array position was the energy difference between site I and the Jth NN1 position with respect to site I; that is, if K were the site number for the Jth NN1 position with respect to site I, then

$$\text{(A-6)} \qquad DE(J,I) = PE(K) - PE(I).$$

Since the crystal was always bound, every PE(I) and PE(K) was negative, which implied net attraction or crystal cohesiveness. However, DE(J,I) may have either positive or negative values, depending on whether site I or site K was more tightly bound. A negative DE(J,I) implied that site K was more tightly bound than site I, while a positive DE(J,I) meant that site I was more tightly bound than site K.

Complications arose since it was unphysical to expect that any atom (occupied site) had no NN1's. No atom was permitted to jump into a site that was already occupied - this implied destruction or loss of matter, which was not allowed. The case of interchanging the atoms between two adjacent occupied sites was physically allowable, but uninteresting since it did not change the shape of the microstate, and so was not considered. Finally, transition from a vacant site was not permitted, since this implied creation or addition of matter which was not considered.

Sites with no NN1's were not encountered in practice (exept for the unphysical "experiment" at 10,000 °K). The problem of jumping from a vacant site and "creating" an atom was easily solved by skipping further consideration of a site if it were vacant. The cases of jumping from one occupied site to another (matter destruction) and of interchanging the atoms in adjacent occupied sites (uninteresting interchange) were taken care of by defining an arbitrarily large DE(J,I) = 1000.0 in the program whenever an occupied site (I) found that the site at its Jth NN1 position was occupied. Later, when transition probabilities were computed, any DE(J,I) ≥ 900.0 was ignored, that is, no transitions were allowed (or in other words, the transition was given a probability of occurrence of 0.0). Sites in the layer $z = z_{max}$ "looking" out the top of the active lattice volume always "saw" site IVAC which was not allowed to be occupied - such sites were also given a DE(J,I) = 1000.0 to avoid atom loss.

Intuitively, physical systems reside at or near the bottom of potential wells, so one expects that for

$$DE(J,I) < 0$$

the atom at site I would probably jump to site K, since it would then have a lower potential. In fact, by the way transition probability was defined, this was exactly what happened. The transition probability, p, was defined as [10]:

$$\text{(A-7a)} \qquad p = 1 \qquad\qquad DE(J,I) \leq 0$$

$$\text{(A-7b)} \qquad p = \exp(-DE(J,I)/kT) \qquad DE(J,I) > 0$$

where k = Boltzmann's constant

T = absolute temperature.

Note that in the program, "kT" was defined at "TAU", that is,

(A-8) $\quad$ TAU = k*T.

For two or more transitions which satisfy condition (A-7a), the computer searched for the deepest well, and carried out the transition. Here, that would mean that for DE(J,I) $\leq$ 0 that the atom at site I jumped to site K. "Carry out the transition" meant that the computer zeroed the occupation index of site I and placed a "1" in the occupation index of site K. In the event that two or more wells had the same (or zero) depth for the deepest nearby well, the computer chose one of the sites by a random number process. This process involved matrix FTEST which was loaded by nesting DO loops at the beginning of the program. Matrix FTEST(I,J) was 12 x 12 and had elements

| | |
|---|---|
| 0.0 | below the diagonal, |
| 1.0 | on the diagonal, |
| FLOAT(I)/FLOAT(J) | above the diagonal. |

When the energy difference was the same for a jump to two or more different sites, the number of sites with the same energy was used as J and a DO loop index used to vary I from 1 to J. A random number was chosen and the FTEST(I,J) values compared with the random number. If the FTEST(I,J) value equalled or exceeded the random number, the transition was made to the site which was the Ith one which had that particular negative energy difference. If the random number exceeded the FTEST(I,J) value, the loop index was incremented and the new FTEST(I,J) value compared with the

same random number, and so on, until one FTEST(I,J) value equalled or exceeded the random number. In any case, a transition was always made for negative (or zero) energy differences.

The computer decided whether or not two energy differences were "the same" by using a small parameter "EPSLON" - a mnemonic for "epsilon". Two energies were considered to be "the same" if they were within EPSLON of each other. In practice, EPSLON = $1.0 \times 10^{-5}$.

For situations in which all DE(J,I) were positive, the transition decisions were based on a random number process. First, the smallest positive energy difference was found. Then a probability for this lowest positive energy difference was computer by formula (A-7b). Next a random number was chosen on the interval (0,1). If the transition probability equalled or exceeded the random number, the computer carried out the transition, and moved on to the next occupied site to repeat the whole process. If the probability did not equal or exceed the random number, the computer found the next lower positive energy difference, calculated a probability, and compared it to a new random number. This process was continued until either a transition was made to a state of higher energy or the site under study ran out of first nearest neighbor vacant sites - in this latter case, no transitions were made to any other site, and the computer moved on to the next occupied site to repeat the procedure for the new site. During the calculation, if two or more sites had the same positive energy difference, these sites were treated sequentially in a manner similar to sites of the same negative energy differences. Positive

site energies were considered "the same" if they were within EPSLON of each other - this was essentially the same procedure used for the negative DE(J,I)'s.

After any transition, the potentials of the sites affected were "patched". The neighbors of the "old site", that is, the previously occupied one, lost an NN1 or an NN2, therefore, PFIV times the proper energy was subtracted from each of these NN1's and NN2's. In addition, the neighbors of the "new site" each gained an NN1 or an NN2, so PFIV times the proper energy was added to each of these NN1's and NN2's.

The question of how appropriate are these probabilities, (A-7a) and (A-7b), may arise. Such probabilities have been used in the literature on crystal growth, and have been shown by statistical arguments to be correct for an infinitely long chain of microstates. According to Leamy and Jackson:

> "averages over a chain of microstates obtained [through the use of these transition probabilities] ... will converge to the equilibrium properties of the system in the limit of infinite chain length."[10]

b). Pseudo-Random Number Generator. The technique used to generate random numbers was a standard one called "pseudo-random number generation by the multiplicative congruential method". The term "psuedo-random" implied that the numbers generated were not really random, and this was, in fact, true. However, such methodsare generally accepted provided they pass certain statistical tests which determine whether the numbers generated are, for all practical purposes, random. The tests require that the numbers generated be uniform on the interval (0,1), and that the sequence

be non-repeating for a sufficiently long period. Other, more involved tests also exist. This method has the advantage that it is reproducible (so that the same probabilities may be regenerated at will as a check or to show the effect of different initial conditions on the final outcome) and fast (this system can work "in line"). The term "congruential" refers to a congruence relation in number theory and modulo arithmetic (which deals only with remainders) [14].

Essentially, the method relied on the fact that the computer (specifically, the IBM 360/67) had a fixed storage length of four bytes (32 binary digits or bits) for integer constants. This meant that the highest integer the computer could store was

$$2^{31} - 1 = 2\ 147\ 483\ 647$$

a ten digit number (allowing the leading bit for sign plus the integer "0", there were $2^{31}$-1 locations/combinations for other integers [3]. If one multiplied together two integers, each with five or six digits, there was generally an overflow of the machine's storage capacity. FORTRAN did not recognize this as an error, and gave only the last eight, nine or ten digits; that is, the leading (and generally most significant) digits were truncated, and the remainder was stored as an integer. Essentially, it was this integer remainder that was the random number. To make it useful, one must convert it to a floating point number on the interval (0,1). This was done by making the integer a floating point number and dividing by $2^{32}$, the modulus of the IBM 360/67. Since about half the integer remainders (on the average) were negative (due to a negative sign

in the sign bit of the integer remainder), one actually had numbers on the interval (-0.5, + 0.5), so one must add 0.5 to obtain numbers on the interval (0,1) [11].

In practice, a total of four FORTRAN statements were required to obtain the first random number. They were, for example:

```
 9 KRAN   = 16807
   IRAN   = 87345
10 IRAN   = IRAN*KRAN
11 RANDOM = 0.5 + FLOAT(IRAN)*2.328306 E-10.
```

The number in exponential form in statement 11 is $2^{-32}$ written as a decimal number. Whenever one needed a new random number, one simply used statements 10 and 11. Note that each random number "RANDOM" was generated from the remainder of the previous integer remainder "IRAN". This system was fast since it worked "in line", and computer literature indicates that this generator can produce about 7,200 random numbers a second [11]. Since this was a pseudo-random generator, its period was finite. However, the theoretical maximum period was about $2^{29}$ or about 537 million numbers [14]. This type pseudo-random generator is regularly used to generate a million or more random numbers. Statistical tests performed on the numbers used and observations of the random numbers produced indicated that the pseudo-random numbers generated were good enough for the practical testing purposes for which they were used here.

### 5. Computer Output

The problem of how to present microstate information in easily interpreted form was solved by having the computer "draw" pictures of microcrystal planes, called "microstate pictures";

that is, the computer printed arrays of zeroes and ones to indicate vacancies or occupied sites, and each digit was in its correct x and y coordinate position for the site it represented. See Figure 23. The program contained the capability of printing out microstate pictures for IX = 10, IX = 20 and IX = 26 (with two planes side by side), and for IX = any even integer through 60. The only restriction on IY and IZ was that they must be even integers. The program also contained the capability of choosing the appropriate output segment to use.

Variable numbers of microstates generated were allowed, and this was provided via input parameter "NUMRUN" - the number of microstates generated. The remainder of the output segment was "nice to have" information that was printed out to avoid looking through the program.

For cases in which large numbers of microstates were generated, it was desired to have only summary information or microstate pictures every mth microstate, and have all microstate pictures printed after a certain number of microstates were generated. This capability was provided by input parameters "MSUM", "MPIX" and "MCRIT". MSUM was the index indicating that a short summary of the current microstate was printed after each MSUM microstates were generated; for example, if MSUM = 5, a summary was printed every fifth microstate. The summary was merely the number of sites occupied in each z-layer. MPIX was the index indicating that a microstate picture was printed after each MPIX microstates were generated; for example, if MPIX = 10, a microstate picture was printed of every tenth microstate. MCRIT was the index

indicating how many microstates were generated prior to printing out pictures of every microstate; for example, if MCRIT = 250, every microstate from 250 to NUMRUN was printed as a picture.

## APPENDIX B

### LATTICE LOADING STATEMENTS

The different initial microstates were generated by sets of one or more program statements inserted just prior to statement 45 in the lattice generator. The statements used for the various lattices generated follow.

1. Perfect Surface

```
IF (M-LL/2)  45,45,46
```

This statement fills the bottom half of the active lattice volume with atoms and leaves the top half empty.

2. Perfect Surface Plus Randomly Placed Extra Atoms

```
   IF (M-LL/2)  45,45,49
49 IF (M-LL/2 - IX*IY/2) 48,48,46
48 IRAN = IRAN * KRAN
   RAND = FLOAT (IRAN) * 2.328306 E-10
   IF (RAND)  45,45,46
```

This set of statements fills the bottom half of the active lattice with atoms, places atoms randomly on the next plane, and leaves all other active lattice planes empty. Sample microstates achieved with this set of statements are sketched as Figures 3a and 4a.

3. Perfect Surface Plus Half Plane with Edge Parallel to X-Axis

```
IF (M - (LL/2 + IX * IY/4)) 45,45,46
```

This statement fills the bottom half of the active lattice with atoms, places a half plane of atoms as a monatomic step in the next plane, and leaves all other active lattice planes empty. The initial microstate achieved is sketched as Figure 5a.

4. Pyramids

```
   IA = 2
   IB = 2
   IF (LZ) 20,45,20
20 IXP = LZ*(IA+1)
   IXM = IX-1-IXP
   IF (LX-IXP) 46,25,25
25 IF (LX-IXM) 30,30,46
30 IYP = LZ*(IB+1)
   IYM = IY-1-IYP
   IF (LY-IYP) 46,35,35
35 IF (LY-IYM) 45,45,46
```

This set of statements loads a pyramid with steps that move in three atomic layers at a time - see Figure 6a. To achieve a pyramid with steps that move in two (one) atomic layers (layer) at a time, change both IA and IB to one (zero) - see Figure 7a.

5. Pyramid holes

```
   IA = 2
   IB = 2
   IF (LZ) 20,45,20
20 IXU = (IX/2) + LZ*(IA+1)-1
   IXL = (IX/2) - LZ*(IA+1)
   IF (LX-IXL) 45,25,25
25 IF (LX-IXU) 30,30,45
30 IYU = (IY/2) + LZ*(IB+1)-1
   IYL = (IY/2) - LZ*(IB+1)
   IF (LY-IYL) 45,35,35
35 IF (LY-IYU) 46,46,45
```

This set of statements loads a pyramid hole with steps that are three atomic layers wide - see Figure 8a. To achieve a pyramid hole with steps two (one) atomic layers (layer) wide, change both IA and IB to one (zero) - see Figure 9a.

6. Ridges

To achieve a ridge parallel to the y-axis, use the same steps as for pyramids, but use the statement

```
IB = -1
```

Step width is determined by parameter lA; for:

IA = 2, steps are three atomic layers wide - see Figure lla;

IA = 1, steps are two atomic layers wide;

IA = 0, steps are one atomic layer wide.

To achieve a ridge parallel to the x-axis, again use the same steps as for pyramids, but use the statement

IA = -1.

Step width is now determined by parameter IB; for

IB = 2, steps are three atomic layers wide - see Figure 10a;

IB = 1, steps are two atomic layers wide;

IB = 0, steps are one atomic layer wide.

7. Valleys Parallel to the X-Axis

```
   IB = 2
   IF (LZ) 30,45,30
30 IYU = (IY/2) + LZ*(IB+1)-1
   IYL = (IY/2) - LZ*(IB+1)
   IF (LY-IYL) 45,35,35
35 IF (LY-IYU) 46,46,45
```

This set of statements loads a valley parallel to the x-axis with a step width of three atomic layers - see Figure 12 a. To achieve such a valley with a step width of two (one) atomic layers (layer), change IB to one (zero) - see Figure 13a.

8. Valleys Parallel to the Y-Axis

```
   IA = 2
   IF (LZ) 20,45,20
20 IXU = (IX/2) + LZ*(IA+1)-1
   IXL = (IX/2) - LZ*(IA+1)
   IF (LX-IXL) 45,25,25
25 IF (LX-IXU) 46,46,45
```

This set of statements loads a valley parallel to the y-axis with a step width of three atomic layers - see Figure 14a. To

achieve such a valley with a step width of two (one) atomic layers (layer), change IA to one (zero) - see Figure 15a.

9. (1 -1 1) Plane

```
IF (LX+LZ-LY) 45,45,46
```

This statement loads a (1 -1 1) plane - see Figure 16a. The plane is truncated if $z_{max}$ is smaller than either $x_{max}$ or $y_{max}$.

10. (-1 1 1) Plane

```
IF (LY+LZ-LX) 45,45,46
```

This statement loads a (-1 1 1) plane - see Figure 17a. This plane is truncated if $z_{max}$ is smaller than either $x_{max}$ or $y_{max}$.

# APPENDIX C

## GLOSSARY OF COMPUTER SYMBOLS

Note: This Appendix also contains the definitions of several terms peculiar to this simulation.

A(J,I): A 4 x 5 array coupled between the MAIN program and Subroutine CROSYM; it carries information for the computation of the coefficients of the cubic potential.

ALPHA: Constant read in as data used in computing parameters for the truncated Morse potential.

BOLTZ: The Boltzmann constant, equal to $8.6171 \times 10^{-5}$ eV/°K.

CARRY OUT A TRANSITION: The simulation procedure used to model a "jump"; the occupation index of the original site is set equal to zero, and the occupation index of the final site (which is an NN1 of the original site) is set equal to one.

CFO: A constant computed by the program for use in the force equation resulting from the cubic potential.

CF1: A constant computed by the program for use in the force equation resulting from the cubic potential.

CF2: A constant computed by the program for use in the force equation resulting from the cubic potential.

CGB1: A constant computed by the program for use in the truncated Morse potential.

CGB2: A constant calculated by the program for use in the truncated Morse potential.

CGD1: A constant calculated by the program for use in the truncated Morse potential.

CGD2: A constant calculated by the program for use in the truncated Morse potential.

CGF1: A constant calculated by the program for use in the force equation resulting from the truncated Morse potential.

CGF2: A constant calculated by the program for use in the force equation resulting from the truncated Morse potential.

COMA: Labeled COMMON storage.

CP0: A constant computed by Subroutine CROSYM for use in the cubic potential.

CP1: A constant computed by Subroutine CROSYM for use in the cubic potential.

CP2: A constant computed by Subroutine CROSYM for use in the cubic potential.

CP3: A constant computed by Subroutine CROSYM for use in the cubic potential.

CROSYM: A Subroutine developed at Lockheed to solve several simultaneous linear equations by the "Method of Crout".

CVD: "Distance conversion factor", CVR x $10^{-10}$ to convert lattice units to meters.

CVDE: CVD/CVE, a ratio to avoid repeated division; the reciprocal of CVED.

CVE: "Energy conversion factor", 1.6 x $10^{-19}$ to convert electronvolts to joules.

CVED: CVE/CVD, a ratio used to avoid repeated division; the reciprocal of CVDE.

CVM: "Mass conversion factor", 1.672 x $10^{-27}$ to convert atomics mass units to kilograms

CVR: "Radial distance conversion factor", the lattice unit in angstroms, used to convert lattice units to angstrom units.

DCON: Constant read/in as data, and used in computing parameters for the truncated Morse potential.

DE(J,I): "Difference in energy", an array containing the energy difference between an atom at site I and its Jth NN1 position, whether occupied or vacant (if occupied, the particular array value is made arbitrarily large to avoid matter destruction); the values on this array are used in determining transition probabilities; this array must be dimensioned at least 12 x LL.

DIS: SQRT(DIST2(I))

DIST(I): "Distance"; a vector array containing the value of the distance (in lattice units) between Ith NN's.

DIST2(I): "Distance squared"; a vector array containing the value of the distance squared (in lattice units squared) between Ith NN's.

EMIN: "Minimum energy"; the value at the lowest DE(J,I), found by searching, when a jump is to be made to a site at lower potential; <u>or</u> the current value of DE(J,I), found by searching and working from lowest to highest energy differences, used when attempting a jump to a site of higher potential.

EMINXT: "Next minimum energy"; the value of the next lower DE(J,I), found by searching, used when attempting to jump to a site of higher potential after all attempts at jumping have failed for lower positive energy differences.

EPSLON: "Epsilon"; a constant used to determine if two DE(J,I)'s were effectively equal; its value is 1.0 x $10^{-5}$ eV.

EQTEMP: "Equivalent temperature"; the effective lattice temperature achieved as a result of varying PFIV; EQTEMP = 0.5 * TEMP/PFIV, and this relation is plotted in Figure 22.

EV: "Electron volt".

EXA: A constant read in as data and used in calculation of the Gibson II potential.

EXB: A constant read in as data and used in calculation of the Gibson II potential.

FLAT: Constant read in as data and equal to 2.0*CVR.

FRCAND(X): "Anderman force"; a function defined in the MAIN program for calculation of the force arising due to the truncated Morse potential.

FRCGIB(X): "Gibson force"; a function defined in the MAIN program for calculation of the force arising due to the Gibson II potential.

FTEST(I,J): A 12 x 12 array used in choosing the site to which an atom jumps if more than one of the NN1 sites of an atom under consideration have the same energy; for different crystal orientations the number of NN1's may change, and therefore the dimensions of array FTEST must also change.

FXA: A constant computed by the program for use in the force equation resulting from the Gibson II potential.

I: A DO loop index.

IA: A constant used by the lattice loader to determine the step width of the initial microstate for several different configurations; see Appendix B.

IB: A constant used by the lattice loader to determine the step width of the initial microstate for several different configurations; see Appendix B.

ICOUNT: A counting index which is the number of the microstate being generated by the program.

IFUL: "Full"; the number of a lattice site in the perfect crystalline substrate on which the active lattice volume is placed; all lattice sites in this substrate are always full and each is represented by site number IFUL.

IH2: Alphanumeric array read in as data for truncated Morse potential parameters.

IHEAD: Alphanumeric array read in as data to provide an output heading for the particular simulation attempted.

IHNO1: Alphanumeric array read in as data to provide a program flag warning that during a calculation an atom was found that had no NN1's (this is unphysical in this simulation); the message merely states "NO NN1's".

IHTARG: Alphanumeric array read in as data and used to define the particular metal simulated.

IHTPOT: Alphanumeric array read in as data to provide a title for Gibson II potential parameters.

II: A DO loop index.

IJ: A DO loop index.

IK: A DO loop index.

IRAN: The present (or initial) value of the constant used in computing pseudo-random numbers.

IRANI: "Initial IRAN"; defined to retain a knowledge of the original random number seed used in a simulation.

ISX: IX/2, a shift index indicating a shift of one row of atoms.

ISXM: ISX-1, a shift index indicating a shift of one atom less than a full row.

ISXP: ISX+1, a shift index indicating a shift of one atom more than a full row.

ISZ: ISX*IY, a shift index indicating a shift of one plane of atoms.

ISZM: ISZ-1, a shift index indicating a shift of one atom less than a full plane.

ISZP: ISZ+1, a shift index indicating a shift of one atom more than a full plane.

IT: An index used in computing ITT.

ITEST: A DO loop index; also, a constant used to determine which site may receive an atom when carrying out a transition in the case of several available sites with the same energy.

ITT: IT+JT+KT, the index which determines when or not a particular coordinate position should be labeled as a valid lattice site by the lattice generator.

IVAC: "Vacant"; the number of a lattice site in the void above the active lattice volume; all sites in this void are always vacant and each is represented by site number IVAC.

IX: An even integer read in as data indicating the number of active lattice volume planes generated in the x-direction.

IXL: A variable used as a shifter when generating pyramid holes and valleys; see Appendix B.

IXM: A variable used as a shifter when generating pyramids and ridges; see Appendix B.

IXP: A variable used as a shifter when generating pyramids and ridges; see Appendix B.

IXU: A variable used as a shifter when generating pyramid holes and valleys; see Appendix B.

IY: An even integer read in as data indicating the number of active volume planes generated in the y-direction.

IYL: A variable used as a shifter when generating pyramid holes and valleys; see Appendix B.

IYM: A variable used as a shifter when generating pyramids and ridges; see Appendix B.

IYP: A variable used as a shifter when generating pyramids and ridges; see Appendix B.

IYU: A variable used as a shifter when generating pyramid holes and valleys; see Appendix B.

IZ: An even integer read in as data indicating the number of active lattice volume planes generated in the z-direction.

J: A DO loop index.

JJ: IY-J, a constant used in the output segment to draw and label microstate pictures.

JJM: JJ-1, a constant used in the output segment to draw and label microstate pictures.

JLOE(I): "Low energy index"; a vector array used to record the site number of the lowest energy NN1 site of an atom attempting to make a transition; this vector must have dimension of at least 12.

JMAX: J-1, an index indicating the number of NN1 sites of lower energy for an atom attempting to make a transition.

JP1: J+1, a constant used in the output segment to draw and label microstate pictures.

JT: An index used in computing ITT.

JUMP: The physical process of an atom leaving its present location and moving into a vacant NN1 site.

KA: K*ISZ, a constant used in the output segment to draw and label microstate pictures.

KB: KA-J*ISZ+1, a constant used in the output segment to draw and label microstate pictures.

KC: KA-JP1*ISX+1, a constant used in the output segment to draw and label microstate pictures.

KD: KB+ISXM, a constant used in the output segment to draw and label microstate pictures.

KE: KC+ISXM, a constant used in the output segment to draw and label microstate pictures.

KF: KP1*ISZ, a constant used in the output segment to draw and label microstate pictures.

KG: KF-J*ISX+1, a constant used in the output segment to draw and label microstate pictures.

KH: KF-JP1*ISX+1, a constant used in the output segment to draw and label microstate pictures.

KI: KG+ISXM, a constant used in the output segment to draw and label microstate pictures.

KJ: KH+ISXM, a constant used in the output segment to draw and label microstate pictures.

KKOUNT: An index used to avoid recalculating potential coefficients when more than one crystal simulation is attempted.

KM1: K-1,an index used in printing out microstate summary information; also, a constant used to draw and label microstate pictures.

KNEW: "New K"; site to which a transition has just been carried out.

KOUNTM: KOUNT2-1, an index used in deciding which site to jump to when there is more than one NN1 site of lowest energy to choose from.

KOUNT1: A counting index used to determine the number of NN1 sites that have lower or higher energy with respect to the site under consideration.

KOUNT2: A counting index used to determine the multiplicity of NN1 sites with lowest energy.

KOUNT3: A counting index used to determine the multiplicity of NN1 sites with higher energy.

KOUNT4: A counting index used to determine whether or not all higher energy sites have been tested for the possibility of making a transition.

KOUNT5: An index controlling program flow when attempting to determine if all sites of higher energy have been tested.

KOUNT6: A counting index indicating the multiplicity of NN1 sites with next higher energy.

KP1: K+1, an index used in printing out microstate summary information; also, a constant used to draw and label microstate pictures.

KP2: K+2, an index used in printing out microstate summary information.

KP3: K+3, an index used in printing out microstate summary information.

KRAN: A constant used in computing pseudo-random numbers.

KT: An index used in computing ITT.

LATTICE UNIT (L.U.): The distance between two NN1's along a single coordinate direction.

LDX: A shift index used to assign lattice site coordinate positions in the x-direction.

LDY: A shift index used to assign lattice site coordinate positions in the y-direction.

LDZ: A shift index used to assign lattice site coordinate positions in the z-direction.

LL: "Lattice length"; the number of available sites in the active lattice volume, equal to IX*IY*IZ/2.

L.U: See LATTICE UNIT.

LX: The x-coordinate position assigned to a particular site during lattice generation.

LY: The *y*-coordinate position assigned to a particular site during lattice generation.

LZ: The *z*-coordinate position assigned to a particular site during lattice generation.

M: A counting index used to assign a label to lattice sites; also, an index (= MCRO) used in Subroutine CROSYM.

MCRIT: A constant read in as data; all microstates generated from microstate number MCRIT to NUMRUN are printed out as microstate pictures.

MCRO: A constant read in as data, indicating the number of unknowns to be solved for by Subroutine CROSYM.

MICROSTATE: The surface configuration either initialized in the program or generated by it.

MICROSTATE PICTURE: An array of occupation indices printed out in their correct coordinate positions to give a physical picture of the surface configuration (see Figure 23).

MICROSTATE SUMMARY: A listing of the microstate number and the number of atoms in each *z*-plane.

MPIX: A constant read in as data indicating that between microstate zero and microstate MCRIT, a microstate picture is printed out every MPIXth microstate.

MSUM: A constant read in as data, indicating that between microstate zero and microstate MCRIT, a microstate summary is printed every MSUMth microstate (except when a microstate picture is printed).

MULT(I): "Multiplicity"; a vector array indicating the numbers of NN1 sites which have the same energy difference with respect to the atom of interest; this vector must have dimension at least as great as the number of NN1's for a lattice location (in this simulation, the proper number was 12).

NA: An index indicating which site an atom jumped to when several NN1 sites of the same energy were located; the value of NA was chosen stochastically.

NBRFOR(J,K): "Neighbor four"; an array containing the site numbers of the 12 NN4's of the Ith lattice site; since this simulation did not use NN4's the array was dimensioned 1 x 1, but if NN4's are included, it must have dimensions of at least 12 x LL.

NBRONE(J,I): "Neighbor one"; an array containing the site numbers of the 12 NN1's of the Ith lattice site; this array must have dimensions of at least 12 x LL.

NBRTHR(J,I): "Neighbor three"; an array containing the site numbers of the 24 NN3's of the Ith lattice site; since this simulation did not use NN3's, the array was dimensioned 1 x 1, but if NN3's are included, it must have dimensions of at least 24 x LL.

NBRTWO(J,I): "Neighbor two"; an array containing the site numbers of the six NN2's of the Ith lattice site; this array must have dimensions of at least 6 x LL.

NCRYST: An index read in as data to indicate the number of crystal simulations to be run using a single input deck; if NCRYST = 0, the simulation with which this value is read in is the last (or the only) simulation run; if NCRYST $\neq$ 0, there is at least one simulation following the one containing a non-zero value of NCRYST.

NN: "Nearest neighbor".

NN1: "First nearest neighbor".

NN2: "Second nearest neighbor".

NNDIS2(I): 'Nearest neighbor distance squared"; the square of the distance to the Ith NN; this vector must have dimension at least as great at NUMNN.

NNFOOC(I): "Number of NN4 sites occupied", defined with respect to site I; this vector must have dimension at least as great as LL.

NNONOC(I): "Number of NN1 sites occupied", defined with respect to site I; this vector must have dimension at least as great as LL.

NNTHOC(I): "Number of NN3 sites occupied", defined with respect to site I; this vector must have dimension at least as great as LL.

NNTWOC(I): "Number of NN2 sites occupied", defined with respect to site I; this vector must have dimension at least as great as LL.

NOCC(I): Vector array containing the occupation index of each lattice site; this vector must have dimension at least as great as LL+3.

NPLANE: "Plane number"; NZ(I)+1, a constant used to define plane numbers greater than zero for use as subscripts.

NUMATM: "Number of atoms"; the number of atoms actually placed in the active lattice volume by the lattice generator.

NUMNN: "Number of nearest neighbors", considered in a particular simulation.

NUMRUN: "Number of runs"; a constant read in as data indicating the number of microstates to be generated by the simulation.

NUMSIT(I); "Number of sites" occupied in the Ith plane; here, I is defined by NPLANE in order to have non-zero subscripts.

NX(I): Vector array containing the values of the x-coordinate for each lattice site; this vector must have dimension at least as great as LL.

NY(I): Vector array containing the values of the y-coordinate for each lattice site; this vector must have dimension at least as great as LL.

NZ(I): Vector array containing the values of the z-coordinate for each lattice site; this vector must have dimension at least as great as LL.

OCCUPATION INDEX: A number indicating whether or not a site is occupied; an occupation index of zero implies that the site is vacant, and an occupation index of one implies that the site is occupied; no other numbers are allowed as an occupation index.

PE(I): "Potential energy"; a vector array containing the value of the potential energy at each lattice site due to nearby occupied sites; this vector must have dimension at least as great as LL.

PENN(I): "Potential energy due to nearest neighbor"; the potential energy contribution due to the Ith NN.

PFIV: "Point five"; an energy distribution factor employed in calculating site potential energy and equivalent lattice temperature; the correct average value of PFIV is 0.5.

POTF(X): A function defined in the MAIN program for calculation of the Gibson II potential.

POT2F(X): A function defined in the MAIN program for calculation of the cubic potential.

POT3F(X): A function defined in the MAIN program for calculation of the truncated Morse potential.

RAND: FLOAT(IRAN)*2.328306E-10, a pseudo-random number on the interval (-0.5, 0.5) used in placing atoms randomly on a perfect surface; see Appendix B.

RANDOM: 0.5+FLOAT(IRAN)*2.328306E-10, a pseudo-random number on the interval (0,1) used for comparison with probabilities when a decision based on a probability was made.

RE: A constant read in as data used in computing truncated Morse potential parameters.

ROEA: A constant read in as data equal to the distance in lattice units at which the Gibson II and cubic potentials match.

ROEB: A constant read in as data equal to the distance in lattice units at which the cubic and truncated Morse potentials match.

ROEC: A constant read in as data equal to the distance in lattice units at which the truncated Morse potential is truncated.

ROEC2: "ROEC squared"; ROEC*ROEC.

SUM: A variable used by Subroutine CROSYM.

TAR: Alphanumeric array read in as data and not used in this simulation.

TAU: BOLTZ*TEMP, i.e, the "kT" factor used in computing a Boltzmann factor.

TEMP: "Temperature"; the nominal lattice temperature; see EQTEMP.

TMAS: "Mass"; a constant read in as data equal to the mass of a lattice atom in atomic mass units.

TPOT: "Total potential"; the sum of all occupied site potentials due to interactions with nearest neighbors through NUMNN.

TPROB: "Transition probability", defined by equations A-7.

COPPER CRYSTAL SURFACE DYNAMICS SIMULATION

```
// EXEC FORTCLG,REGION=190K
//FORT.SYSIN DD *
C     THIS PROGRAM CONTAINS A NON-STANDARD USE OF THE ENERGY DISTRIBUTION
C     FACTOR, "PFIV"
      DIMENSION NX( 800),NY( 800),NZ( 800),NOCC( 805),NNDIS2( 4),
     1PENN( 4),PE( 800),FTEST(12,12),JLOE(12),DE(12, 800),DIST2( 4),
     2MULT(12),NBRONE(12, 800),NNONOC( 800),NUMSIT(  30),
     3NBRTWO( 6, 800),NNTWOC( 800),NBRTHR( 1,   1),NNTHOC( 800),
     4NBRFOR( 1,   1),NNFOOC( 800)
      DIMENSION IHTARG(2),IHTPOT(2),IH2(10)
      DIMENSION IHNO1(3),IHEAD(20)
      DIMENSION DIST(4)
      COMMON/COMA/ A(4,5),MCRO
      POTF(X)=EXP(EXA+EXB*X)
      POT2F(X)=X*(X*(X*CP3+CP2)+CP1)+CP0
      POT3F(X)=EXP(CGD1+CGB1*X)-EXP(CGD2+CGB2*X)
      FRCGIB(X)=EXP(FXA+EXB*X)*CVDE
      FRCAND(X)=(EXP(CGF1+CGB1*X)-EXP(CGF2+CGB2*X))*CVDE
 9000 FORMAT (20A4)
 9010 FORMAT (4I5)
 9015 FORMAT (5I5)
 9020 FORMAT (F10.5,I5)
 9130 FORMAT(2A4,A4,F8.3,2A4,2F10.5,F8.5)
 9140 FORMAT(10A4,3F5.2,3F8.5,I1)
 9210 FORMAT (3A4)
 9220 FORMAT (2I5,3A4)
 9230 FORMAT (2I10//)
 9240 FORMAT (2I6,F14.7)
 9300 FORMAT (T 3,4(3X,'I',3X,'X',3X,'Y',3X,'Z',3X,'PE(I)',3X,'OCC')/)
 9310 FORMAT (' ',4(I5,3I4,F10.6,I3))
 9340 FORMAT (' ',/T5,'Y',T65,'Y')
 9360 FORMAT (' ',T7,'Z =',I3,' PLANE',T67,'Z =',I4,' PLANE')
 9370 FORMAT (' ',T7,'NUMBER OF SITES OCCUPIED = ',I3,T67,'NUMBER OF SIT
     1ES OCCUPIED = ',I3/)
C
C     THE NEXT FIVE FORMATS ARE USED IN PRINTING MICROSTATES WITH IX = 10
C
 9320 FORMAT (' ',I2,' +',5(3X,I1), ' +',T62,I2,' + ' ,5(I1,3X),'+')
 9330 FORMAT (' ',I2,' + ' ,5(I1,3X),'+',T62,I2,' +',5(3X,I1), ' +')
 9345 FORMAT (' ',T5,'+',11( ' +'),T65,'+',11( ' +'))
 9350 FORMAT (' ',T5,'+',11( ' +'), ' X',T65,'+',11( ' +'), ' X')
 9355 FORMAT(' ', T7,'0   2   4   6   8',T67,'0   2   4   6   8'/)
C
C     THE NEXT FIVE FORMATS ARE USED IN PRINTING MICROSTATES WITH IX = 20
```

```
C
 9321 FORMAT (' ',I2,' +',10(3X,I1),' +',T62,I2,' + ',10(I1,3X),'+')
 9331 FORMAT (' ',I2,' + ',10(I1,3X),'+',T62,I2,' +',10(3X,I1),' +')
 9346 FORMAT (' ',T5,'+',21(' +'),T65,'+',21(' +'))
 9351 FORMAT (' ',T5,'+',21(' +'),' X',T65,'+',21(' +'),' X')
 9356 FORMAT (' ',T7,'0   2   4   6   8  10  12  14  16  18',T67,'0   2
     1 4   6   8  10  12  14  16  18'/)
C
C     THE NEXT FIVE FORMATS ARE USED IN PRINTING MICROSTATES WITH IX = 26
C
 9322 FORMAT (' ',I2,' +',13(3X,I1),' +',T62,I2,' + ',13(I1,3X),'+')
 9332 FORMAT (' ',I2,' + ',13(I1,3X),'+',T62,I2,' +',13(3X,I1),' +')
 9347 FORMAT (' ',T5,'+',27(' +'),T65,'+',27(' +'))
 9352 FORMAT (' ',T5,'+',27(' +'),' X',T65,'+',27(' +'),' X')
 9357 FORMAT (' ',T7,'0   2   4   6   8  10  12  14  16  18',
     1'  20  22  24',T67,'0   2   4   6   8  10  12  14  16',
     2'  18  20  22  24'/)
C
C     THE NEXT 8 FORMATS ARE USED IN PRINTING MICROSTATE
C     PICTURES  FOR IX .NE. 10 .OR. 20 .OR. 26, BUT .LE. 60 .AND. EVEN
C
 9341 FORMAT (' ',/T5,'Y')
 9323 FORMAT (' ',I2,' +',30(3X,I1),' +')
 9333 FORMAT (' ',I2,' + ',30(I1,3X),'+')
 9348 FORMAT (' ',T5,'+',61(' +'))
 9353 FORMAT (' ',T5,'+',61(' +'),' X')
 9358 FORMAT (' ',T7,'0   2   4   6   8  10  12  14  16',
     1'  18  20  22  24  26  28  30  32  34  36  38  40',
     2'  42  44  46  48  50  52  54  56  58'/)
 9361 FORMAT (' ',T7,'Z =',I3,' PLANE')
 9371 FORMAT (' ',T7,'NUMBER OF SITES OCCUPIED = ',I3/)
C
C     THE REMAINING FORMATS ARE USED TO PRINT OUT TITLE
C     AND NICE TO HAVE INFORMATION
C
 9500 FORMAT ('1',T32,'FACE-CENTERED CUBIC CRYSTAL SURFACE',
     1' DYNAMICS, (100) ORIENTATION'///)
 9505 FORMAT (' ',T20,20A4///)
 9510 FORMAT (' ','THE METAL VISUALIZED IS COPPER, ',
     1'USING THE ANDERMAN NN2 POTENTIAL'/)
 9520 FORMAT (' ','MICROCRYSTAL SIZE: X = ',I2,5X,'Y = ',I2,
     15X,'Z = ',I2/)
 9530 FORMAT (' ','NUMBER OF ACTIVE LATTICE SITES = ',I5,
     1T60,'NUMBER OF ATOMS = ',I5/)
 9540 FORMAT (' ','NUMBER OF NEAREST NEIGHBORS USED IN ',
     1'THIS CALCULATION IS ',I2/)
 9550 FORMAT (' ','EQUIVALENT TEMPERATURE = ',F 8.1,' K',
     1T60,'NUMBER OF MICROSTATES GENERATED = ',I4/)
```

```
9560 FORMAT (' ','MICROSTATE SUMMARY EVERY ',I3,' MICRO',
    1'STATES',T60,'MICROSTATE PICTURES EVERY ',I3,' MICRO',
    2'STATES'/)
9570 FORMAT (' ','MICROSTATE ZERO IS THE INITIAL MICRO',
    1'STATE'//)
9580 FORMAT (' ',/' MICROSTATE NUMBER ',I4,' - PICTURE',
    1' FOLLOWS')
9590 FORMAT (' ', 'MICROSTATE NUMBER ', I4,' - SUMMARY',
    1' FOLLOWS'/)
9600 FORMAT (' ', T5,'Z',T10,'SITES OCCUPIED',T35,'Z',T40,
    1'SITES OCCUPIED',T65,'Z',T70,'SITES OCCUPIED',T95,'Z',
    2T100,'SITES OCCUPIED'/)
9610 FORMAT (' ',T4,I2,T13,I4,T34,I2,T43,I4,T64,I2,
    1T73,I4,T94,I2,T103,I4)
9620 FORMAT (/T2,'RANDOM NUMBER INFORMATION:  KRAN = ',
    1I12,';  INITIAL IRAN = ',I12/)
9625 FORMAT (' ','LATEST RANDOM NUMBER INFORMATION:  ',
    1'IRAN = ',I12,'; RANDOM = ',F9.7///)
9630 FORMAT (' ','POTENTIAL/FORCE PARAMETERS:'/)
9640 FORMAT (' ',T10,'GIBSON POTENTIAL:',T30,'EXA = ',
    1F10.5,T50,'EXB = ',F10.5/)
9650 FORMAT (' ',T10,'GIBSON FORCE:',T30,'FXA = ',F10.5/)
9670 FORMAT (' ',T10,'GIBSON-CUBIC BOUNDARY:  ROEA = ',
    1F5.2,' LATTICE UNITS'/)
9680 FORMAT (' ',T10,'CUBIC POTENTIAL:'T30,'CPO = ',F10.4,
    1T50,'CP1 = ',F10.4,T70,'CP2 = ',F10.4,T90,'CP3 = ',F10.4/)
9690 FORMAT (' ',T10,'CUBIC FORCE:',T30,'CFO = ',F12.9,
    1T50,'CF1 = ',F12.9,T70,'CF2 = ',F12.9/)
9700 FORMAT (' ',T10,'CUBIC-ANDERMAN BOUNDARY:  ROEB = ',
    1F5.2,' LATTICE UNITS'/)
9710 FORMAT (' ',T10,'ANDERMAN POTENTIAL:',T30,'CGD1 = ',
    1F10.6,T50,'CGD2 = ',F10.6,T70,'CGB1 = ',F10.6,
    2T90,'CGB2 = ',F10.6/)
9720 FORMAT (' ',T10,'ANDERMAN FORCE:',T30,'CGF1 = ',F10.6,
    1T50,'CGF2 = ',F10.6/)
9730 FORMAT (' ',T10,'ANDERMAN CUTOFF:  ROEC = ',F5.2,
    1' LATTICE UNITS'/)
9740 FORMAT (' ',T10,'NEAREST NEIGHBOR DISTANCES: '/,T21,
    1'DIST(1) = ',F6.3,' LATTICE UNITS',T61,'DIST(2) = ',
    2F6.3,' LATTICE UNITS'/,T21,'DIST(3) = ',F6.3,
    3F6.3,' LATTICE UNITS',T61,'DIST(4) = ',F6.3,
    4' LATTICE UNITS')
9750 FORMAT (/   T11,'POTENTIALS AT NEAREST NEIGHBOR ',
    1'DISTANCES:'/,T16,'PENN(1) = ',F8.6,' EV',T41,
    2'PENN(2) = ',F8.6,' EV',T66,'PENN(3) = ',F8.6,' EV',
    3T91,'PENN(4) = ',F8.6,' EV')
9760 FORMAT (' ','ALL MICROSTATES FROM ',I4,' ON PRINTED'/)
C
```

```
      READ (5,9130) IHTARG,TAR,TMAS,IHTPOT,EXA,EXB,FLAT
      READ (5,9140) IH2,ROEA,ROEB,ROEC,DCON,ALPHA,RE,MCRO
      READ (5,9210) IHNO1
      EPSLON=1.0E- 5
      TEMP=1000.0
      BOLTZ=8.6171E- 5
      TAU=BOLTZ*TEMP
C
C     DEFINE PARAMETERS FOR RANDOM NUMBER GENERATION
C
      KRAN = 16807
      IRAN=87345
C
C     THIS SEGMENT LOADS MATRIX "FTEST" FOR USE IN THE TRANSITION PROBABILITY
C     PORTION OF THE PROGRAM
C
      DO 1 J=1,12
      DO 1 I=1,12
      IF(J-I) 3,4,5
    3 FTEST(I,J)=0.0
      GO TO 1
    4 FTEST(I,J)=1.0
      GO TO 1
    5 FTEST(I,J)=FLOAT(I)/FLOAT(J)
    1 CONTINUE
      KKOUNT = 0
    9 ICOUNT = -1
      READ (5,9000) IHEAD
      READ (5,9010) IX,IY,IZ,LL
      READ (5,9015) MPIX,MSUM,MCRIT,NUMRUN,NCRYST
      READ (5,9020) PFIV,NUMNN
      EQTEMP = 0.5*TEMP/PFIV
      IRANI=IRAN
C
C     THIS IS A LATTICE GENERATOR FOR THE (100) ORIENTATION, FCC CRYSTAL       10
C
      LDX=1
      LDY=1
      LDZ=1
      M=1
      KT=0
      LZ=-LDZ
      DO 50 IK=1,IZ
      LZ=LZ+LDZ
      JT=0
      LY=-LDY
      DO 60 IJ=1,IY
      LY=LY+LDY
```

```
      IT=0
      LX=-LDX
      DO 40 II=1,IX
      LX=LX+LDX
      ITT=IT+JT+KT
      IF(ITT-(ITT/2)*2) 43,47,43
   47 NX(M)=LX
      NY(M)=LY
      NZ(M)=LZ
C     THE NEXT CARD STARTS THE SPECIFIC STEPS TO LOAD AN ACTIVE
C     LATTICE OF A SPECIFIC TYPE
C     LOAD PERFECT SURFACE + HALF PLANE MONATOMIC STEP PARALLEL TO X-AXIS
      IF(M-(LL/2+IX*IY/4)) 45,45,46
C     THIS ENDS THE SPECIFIC LATTICE LOADER
   45 NOCC(M)=1
      GO TO 44
   46 NOCC(M)=0
   44 CONTINUE
      M=M+1
   43 CONTINUE
      IT=IT+1
   40 CONTINUE
      JT=JT+1
   60 CONTINUE
      KT=KT+1
   50 CONTINUE
C
C     SITE IFUL IS ALWAYS FULL, SITE IVAC IS ALWAYS EMPTY - INITIALIZE THEM
C
      IFUL=M+1
      IVAC=M+2
      NOCC(IFUL)=1
      NOCC(IVAC)=0
C
C     NN1 ASSIGNMENT ROUTINE - REQUIRES THAT IX, IY, & IZ ALL BE EVEN
C     DEFINE SHIFTERS
C
      ISX=IX/2
      ISXP=ISX+1
      ISXM=ISX-1
      ISZ=ISX*IY
      ISZP=ISZ+1
      ISZM=ISZ-1
      DO 100 I=1,LL
      IF(NX(I)-(NX(I)/2)*2) 102,101,102
C
C     X VALUE IS EVEN
C
```

```
  101 NBRONE(1,I)=I-ISZP
      NBRONE(2,I)=I+ISZM
      NBRONE(3,I)=I-ISZ
      NBRONE(4,I)=I+ISZ
      NBRONE(5,I)=I-ISXP
      NBRONE(6,I)=I-ISX
      NBRONE(7,I)=I+ISXM
      NBRONE(8,I)=I+ISX
      GO TO 103
C
C     X VALUE IS ODD
C
  102 NBRONE(1,I)=I-ISZ
      NBRONE(2,I)=I+ISZ
      NBRONE(3,I)=I-ISZM
      NBRONE(4,I)=I+ISZP
      NBRONE(5,I)=I-ISX
      NBRONE(6,I)=I-ISXM
      NBRONE(7,I)=I+ISX
      NBRONE(8,I)=I+ISXP
C
C     X VALUE EITHER EVEN OR ODD HEREAFTER - IE, FOR NEXT 4 STATEMENTS
C
  103 NBRONE(9,I)=I-ISZ-ISX
      NBRONE(10,I)=I-ISZ+ISX
      NBRONE(11,I)=I+ISZ-ISX
      NBRONE(12,I)=I+ISZ+ISX
  100 CONTINUE
      DO 195 I=1,LL
C
C     THERE ARE 22 NN1 SPECIAL CASES (SIDES, EDGES, CORNERS) - TEST FOR THEM
C
      IF(NX(I)) 105,110,105
  105 IF(NY(I)) 106,115,106
  106 IF(NZ(I)) 107,120,107
  107 IF(NX(I)-IX+1) 108,125,108
  108 IF(NY(I)-IY+1) 109,130,109
  109 IF(NZ(I)-IZ+1) 195,139,195
C
C     SPECIAL CASE: X=0, LEFT FACE; TEST FOR EDGES
C
  110 IF(NY(I)) 111,140,111
  111 IF(NY(I)-IY+1) 112,144,112
  112 IF(NZ(I)) 113,142,113
  113 IF(NZ(I)-IZ+1) 114,146,114
C
C     LEFT FACE, NOT AN EDGE
C
```

```
  114 NBRONE(1,I)=NBRONE(1,I)+ISX
      NBRONE(2,I)=NBRONE(2,I)+ISX
      NBRONE(5,I)=NBRONE(5,I)+ISX
      NBRONE(7,I)=NBRONE(7,I)+ISX
      GO TO 195
C
C     SPECIAL CASE: Y=0, FRONT FACE; TEST FOR EDGES
C
  115 IF(NZ(I)) 116,148,116
  116 IF(NZ(I)-IZ+1) 118,150,118
  118 IF(NX(I)-IX+1) 119,152,119
C
C     FRONT FACE, NOT AN EDGE
C
  119 NBRONE( 5,I)=NBRONE( 5,I)+ISZ
      NBRONE( 6,I)=NBRONE( 6,I)+ISZ
      NBRONE( 9,I)=NBRONE( 9,I)+ISZ
      NBRONE(11,I)=NBRONE(11,I)+ISZ
      GO TO 195
C
C     SPECIAL CASE: Z=0, BOTTOM FACE; TEST FOR EDGES
C
  120 IF(NX(I)-IX+1) 123,154,123
  123 IF(NY(I)-IY+1) 124,156,124
C
C     BOTTOM FACE, NOT AN EDGE
C
  124 NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      GO TO 195
C
C     SPECIAL CASE: X=IX-1, RIGHT FACE; TEST FOR EDGES
C
  125 IF(NY(I)-IY+1) 128,158,128
  128 IF(NZ(I)-IZ+1) 129,160,129
C
C     RIGHT FACE, NOT AN EDGE
C
  129 NBRONE(3,I)=NBRONE(3,I)-ISX
      NBRONE(4,I)=NBRONE(4,I)-ISX
      NBRONE(6,I)=NBRONE(6,I)-ISX
      NBRONE(8,I)=NBRONE(8,I)-ISX
      GO TO 195
C
C     SPECIAL CASE: Y=IY-1, BACK FACE; TEST FOR REMAINING EDGE
C
```

```
  130 IF(NZ(I)-IZ+1) 134,162,134
C
C     BACK FACE, NOT AN EDGE
C
  134 NBRONE( 7,I)=NBRONE( 7,I)-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISZ
      NBRONE(10,I)=NBRONE(10,I)-ISZ
      NBRONE(12,I)=NBRONE(12,I)-ISZ
      GO TO 195
C
C     SPECIAL CASE: Z=IZ-1, TOP FACE; NO REMAINING EDGES
C
  139 NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      GO TO 195
C
C     SPECIAL  CASE: X=Y=0, LEFT FRONT EDGE; TEST FOR BOTTOM LEFT FRONT CORNER
C     NOTE THAT TOP LEFT FRONT CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Z PLANES
C
  140 IF(NZ(I)) 141,170,141
C
C     LEFT FRONT EDGE, NOT A CORNER
C
  141 NBRONE( 1,I)=NBRONE( 1,I)+ISX
      NBRONE( 2,I)=NBRONE( 2,I)+ISX
      NBRONE( 7,I)=NBRONE( 7,I)+ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISX+ISZ
      NBRONE( 6,I)=NBRONE( 6,I)+ISZ
      NBRONE( 9,I)=NBRONE( 9,I)+ISZ
      NBRONE(11,I)=NBRONE(11,I)+ISZ
      GO TO 195
C
C     SPECIAL CASE: X=Z=0, LEFT BOTTOM EDGE; TEST FOR BOTTOM LEFT FRONT CORNER
C     NOTE-BOTTOM LEFT REAR CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Y PLANES
C     NOTE-BOTTOM LEFT FRONT CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  142 IF(NY(I)) 143,195,143
C
C     BOTTOM LEFT EDGE, NOT A CORNER
C
  143 NBRONE( 2,I)=NBRONE( 2,I)+ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISX
      NBRONE( 7,I)=NBRONE( 7,I)+ISX
      NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
```

```
      NBRONE(10,I)=IFUL
      GO TO 195
C
C     SPECIAL CASE: X=0, Y=IY-1, LEFT BACK EDGE; TEST FOR TOP LEFT BACK CORNER
C     NOTE-BOTTOM LEFT BACK CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Y PLANES
C
  144 IF(NZ(I)-IZ+1) 145,175,145
C
C     LEFT BACK EDGE, NOT A CORNER
C
  145 NBRONE( 1,I)=NBRONE( 1,I)+ISX
      NBRONE( 2,I)=NBRONE( 2,I)+ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISX
      NBRONE( 7,I)=NBRONE( 7,I)+ISX-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISZ
      NBRONE(10,I)=NBRONE(10,I)-ISZ
      NBRONE(12,I)=NBRONE(12,I)-ISZ
      GO TO 195
C
C     SPECIAL CASE: X=0, Z=IZ-1, TOP LEFT EDGE; TEST FOR TOP LEFT BACK CORNER
C     NOTE-TOP LEFT FRONT CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Z PLANES
C     NOTE-TOP LEFT BACK CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  146 IF(NY(I)-IY+1) 147,195,147
C
C     TOP LEFT EDGE, NOT A CORNER
C
  147 NBRONE( 1,I)=NBRONE(1,I)+ISX
      NBRONE( 5,I)=NBRONE(5,I)+ISX
      NBRONE( 7,I)=NBRONE(7,I)+ISX
      NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      GO TO 195
C
C     SPECIAL CASE: Y=Z=0, FRONT BOTTOM EDGE; TEST FOR BOTTOM LEFT FRONT CORNER
C     NOTE-BOTTOM RIGHT FRONT CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF X PLANES
C     NOTE-BOTTOM LEFT FRONT CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  148 IF(NX(I)) 149,195,149
C
C     FRONT BOTTOM EDGE, NOT A CORNER
C
  149 NBRONE( 5,I)=NBRONE( 5,I)+ISZ
      NBRONE( 6,I)=NBRONE( 6,I)+ISZ
      NBRONE(11,I)=NBRONE(11,I)+ISZ
      NBRONE( 1,I)=IFUL
```

```
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      GO TO 195
C     SPECIAL CASE: Y=0,Z=IZ-1, FRONT TOP EDGE; TEST FOR RIGHT FRONT TOP CORNER
C
  150 IF(NX(I)-IX+1) 151,180,151
C
C     FRONT TOP EDGE, NOT A CORNER
C
  151 NBRONE( 5,I)=NBRONE(5,I)+ISZ
      NBRONE( 6,I)=NBRONE(6,I)+ISZ
      NBRONE( 9,I)=NBRONE(9,I)+ISZ
      NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      GO TO 195
C
C     SPECIAL CASE: Y=0,X=IX-1,RIGHT FRONT EDGE; TEST FOR TOP RIGHT FRONT CORNER
C     NOTE-RIGHT FRONT BOTTOM CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF X PLANES
C     NOTE-TOP RIGHT FRONT CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  152 IF(NZ(I)-IZ+1) 153,195,153
C
C     RIGHT FRONT EDGE, NOT A CORNER
C
  153 NBRONE( 3,I)=NBRONE( 3,I)-ISX
      NBRONE( 4,I)=NBRONE( 4,I)-ISX
      NBRONE( 8,I)=NBRONE( 8,I)-ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISZ
      NBRONE( 6,I)=NBRONE( 6,I)+ISZ-ISX
      NBRONE( 9,I)=NBRONE( 9,I)+ISZ
      NBRONE(11,I)=NBRONE(11,I)+ISZ
      GO TO 195
C
C     SPECIAL CASE:Z=0,X=IX-1,RIGHT BOTTOM EDGE;TEST FOR BACK RIGHT BOTTOM CORNE
C     NOTE-RIGHT FRONT BOTTOM CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF X PLANES
C
  154 IF(NY(I)-IY+1) 155,185,155
C
C     RIGHT BOTTOM EDGE, NOT A CORNER
C
  155 NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      NBRONE(4,I)=NBRONE(4,I)-ISX
```

```
      NBRONE(6,I)=NBRONE(6,I)-ISX
      NBRONE(8,I)=NBRONE(8,I)-ISX
      GO TO 195
C
C     SPECIAL CASE:Z=0,Y=IY-1,BOTTOM BACK EDGE;TEST FOR RIGHTBOTTOMBACK CORNER
C     NOTE-LEFT BOTTOM BACK CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Y PLANES
C     NOTE-RIGHT BOTTOM BACK CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  156 IF(NX(I)-IX+1) 157,195,157
C
C     BOTTOM BACK EDGE, NOT A CORNER
C
  157 NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      NBRONE( 7,I)=NBRONE( 7,I)-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISZ
      NBRONE(12,I)=NBRONE(12,I)-ISZ
      GO TO 195
C
C     SPECIAL CASE:X=IX-1,Y=IY-1,BACK RIGHT EDGE;TEST FOR BOTTOMRIGHTBACK CORNER
C     NOTE-TOP RIGHT BACK CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Z PLANES
C     NOTE-BOTTOM RIGHT BACK CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  158 IF(NZ(I)) 159,195,159
C
C     BACK RIGHT EDGE, NOT A CORNER
C
  159 NBRONE( 3,I)=NBRONE( 3,I)-ISX
      NBRONE( 4,I)=NBRONE( 4,I)-ISX
      NBRONE( 6,I)=NBRONE( 6,I)-ISX
      NBRONE( 8,I)=NBRONE( 8,I)-ISX-ISZ
      NBRONE( 7,I)=NBRONE( 7,I)-ISZ
      NBRONE(10,I)=NBRONE(10,I)-ISZ
      NBRONE(12,I)=NBRONE(12,I)-ISZ
      GO TO 195
C
C     SPECIAL CASE:X=IX-1,Z=IZ-1,TOP RIGHT EDGE;TEST FOR FRONT RIGHT TOP CORNER
C     NOTE-BACK RIGHT TOP CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Z PLANES
C     NOTE-FRONT RIGHT TOP CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  160 IF(NY(I)) 161,195,161
C
C     TOP RIGHT EDGE, NOT A CORNER
C
  161 NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
```

```
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      NBRONE(3,I)=NBRONE(3,I)-ISX
      NBRONE(6,I)=NBRONE(6,I)-ISX
      NBRONE(8,I)=NBRONE(8,I)-ISX
      GO TO 195
C
C     SPECIAL CASE:Y=IY-1,Z=IZ-1,TOP BACK EDGE;TEST FOR TOP BACK LEFT CORNER
C     NOTE-TOP BACK RIGHT CORNER CAN'T OCCUR FOR AN EVEN NUMBER OF Z PLANES
C     NOTE-TOP BACK LEFT CORNER ALREADY CORRECTED, SO EXIT IF WE'RE THERE
C
  162 IF(NX(I)) 163,195,163
C
C     TOP BACK EDGE, NOT A CORNER
C
  163 NBRONE( 7,I)=NBRONE( 7,I)-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISZ
      NBRONE(10,I)=NBRONE(10,I)-ISZ
      NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      GO TO 195
C
C     SPECIAL CASE: X=Y=Z=0, BOTTOM LEFT FRONT CORNER
C
  170 NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      NBRONE( 2,I)=NBRONE( 2,I)+ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISX+ISZ
      NBRONE( 6,I)=NBRONE( 6,I)+ISZ
      NBRONE( 7,I)=NBRONE( 7,I)+ISX
      NBRONE(11,I)=NBRONE(11,I)+ISZ
      GO TO 195
C
C     SPECIAL CASE:X=0, Y=IY-1, Z=IZ-1, TOP LEFT BACK CORNER
C
  175 NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      NBRONE( 1,I)=NBRONE( 1,I)+ISX
      NBRONE( 5,I)=NBRONE( 5,I)+ISX
      NBRONE( 7,I)=NBRONE( 7,I)+ISX-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISZ
      NBRONE(10,I)=NBRONE(10,I)-ISZ
```

```
      GO TO 195
C
C     SPECIAL CASE: X=IX-1, Y=0, Z=IZ-1, TOP RIGHT FRONT CORNER
C
  180 NBRONE( 2,I)=IVAC
      NBRONE( 4,I)=IVAC
      NBRONE(11,I)=IVAC
      NBRONE(12,I)=IVAC
      NBRONE(3,I)=NBRONE(3,I)-ISX
      NBRONE(5,I)=NBRONE(5,I)+ISZ
      NBRONE(6,I)=NBRONE(6,I)+ISZ-ISX
      NBRONE(8,I)=NBRONE(8,I)-ISX
      NBRONE(9,I)=NBRONE(9,I)+ISZ
      GO TO 195
C
C     SPECIAL CASE: X=IX-1, Y=IY-1, Z=0, BACK RIGHT BOTTOM CORNER
C
  185 NBRONE( 1,I)=IFUL
      NBRONE( 3,I)=IFUL
      NBRONE( 9,I)=IFUL
      NBRONE(10,I)=IFUL
      NBRONE( 4,I)=NBRONE( 4,I)-ISX
      NBRONE( 6,I)=NBRONE( 6,I)-ISX
      NBRONE( 7,I)=NBRONE( 7,I)-ISZ
      NBRONE( 8,I)=NBRONE( 8,I)-ISX-ISZ
      NBRONE(12,I)=NBRONE(12,I)-ISZ
  195 CONTINUE
C
C    .NN2 ASSIGNMENT ROUTINE - REQUIRES THAT IX, IY, IZ ALL BE EVEN
C
      DO 201 I=1,LL
      DO 200 J=1,6
      GO TO (210,220,230,240,250,260),J
  210 IF (NX(I)-1)  213,213,217
  213 NBRTWO(J,I)=I+IX/2-1
      GO TO 200
  217 NBRTWO(J,I)=I-1
      GO TO 200
  220 IF(NX(I)-IX+2) 223,227,227
  223 NBRTWO(J,I)=I+1
      GO TO 200
  227 NBRTWO(J,I)=I-IX/2+1
      GO TO 200
  230 IF (NY(I)-1) 233,233,237
  233 NBRTWO(J,I)=I+(IX/2)*(IY-2)
      GO TO 200
  237 NBRTWO(J,I)=I-IX
      GO TO 200
```

```
  240 IF(NY(I)-IY+2) 243,247,247
  243 NBRTWO(J,I)=I+IX
      GO TO 200
  247 NBRTWO(J,I)=I-(IX/2)*(IY-2)
      GO TO 200
  250 IF(NZ(I)-1) 253,253,257
  253 NBRTWO(J,I)= IFUL
      GO TO 200
  257 NBRTWO(J,I)=I-IX*IY
      GO TO 200
  260 IF (NZ(I)-IZ+2) 263,267,267
  263 NBRTWO(J,I)=I+IX*IY
      GO TO 200
  267 NBRTWO(J,I)=IVAC
  200 CONTINUE
  201 CONTINUE
C
C     THIS COMPUTES THE POTENTIAL ENERGY OF EACH LATTICE SITE (WHETHER
C       OCCUPIED OR NOT) WITH RESPECT TO NEAREST NEIGHBOR OCCUPIED
C       SITES THROUGH "NUMNN"
C
      IF (KKCOUNT) 1202,1000,1202
 1000 CVR = 0.5*FLAT
      CVE=1.60E-19
      CVM=1.672E-27
      CVD=CVR*1.0E-10
      CVDE=CVD/CVE
      CVED=CVE/CVD
      FXA=ALOG(-EXB*CVED)+EXA
      ROEC2=ROEC*ROEC
      CGD1=ALOG(DCON)+2.0*ALPHA*RE
      CGD2=ALOG(2.0*DCON)+ALPHA*RE
      CGB1=-2.0*ALPHA*CVR
      CGB2=-ALPHA*CVR
      CGF1=ALOG(-CGB1*CVED)+CGD1
      CGF2=ALOG(-CGB2*CVED)+CGD2
      A(1,1)=1.0
      A(1,2)=ROEA
      A(1,3)=ROEA*ROEA
      A(1,4)=ROEA*ROEA*ROEA
      A(1,5)=POTF(ROEA)
      A(2,1)=1.0
      A(2,2)=ROEB
      A(2,3)=ROEB*ROEB
      A(2,4)=ROEB*ROEB*ROEB
      A(2,5)=POT3F(ROEB)
      A(3,1)=0.0
      A(3,2)=-1.0
```

```
      A(3,3)=-2.0*ROEA
      A(3,4)=-3.0*ROEA*ROEA
      A(3,5)=FRCGIB(ROEA)
      A(4,1)=0.0
      A(4,2)=-1.0
      A(4,3)=-2.0*ROEB
      A(4,4)=-3.0*ROEB*ROEB
      A(4,5)=FRCAND(ROEB)
      CALL CROSYM
      CP0=A(1,5)
      CP1=A(2,5)
      CP2=A(3,5)
      CP3=A(4,5)
      CF0=-CP1*CVED
      CF1=-2.0*CP2*CVED
      CF2=-3.0*CP3*CVED
C
C     THIS "DO" LOOP SETS VALUES OF THE SQUARE OF THE DISTANCE FOR THE NEAREST
C     NEIGHBOR NUMBER INDICATED BY THE LOOP INDEX - THIS FORMULA IS EXACT FOR
C     NN1 THROUGH NN13 FOR THE FCC CRYSTAL, 100 ORIENTATION, IN LATTICE UNITS
C
 1050 DO 1100 N=1,NUMNN
      NNDIS2(N)=2*N
 1100 CONTINUE
C
C     THIS "DO" LOOP FINDS THE POTENTIAL ENERGY OF NEAREST NEIGHBORS
C     THROUGH "NUMNN"; NOTE THAT THE POTENTIAL ENERGY FORMULAS ARE
C     WRITTEN FOR DISTANCES IN LATTICE UNITS
C
      DO  1200 N=1,NUMNN
      DIST2(N)=FLOAT(NNDIS2(N))
      DIST(N)=SQRT(DIST2(N))
      IF(DIST2(N)-ROEC2) 1210,1290,1290
 1210 DIS =SQRT(DIST2(N))
      IF (DIS -ROEA) 1220,1230,1230
 1220 PENN(N)=POTF(DIS )
      GO TO 1200
 1230 IF (DIS -ROEB) 1240,1250,1250
 1240 PENN(N)=POT2F(DIS )
      GO TO 1200
 1250 PENN(N)=POT3F(DIS )
      GO TO 1200
 1290 PENN(N)=0.0
 1200 CONTINUE
      KKOUNT = KKOUNT +1
 1202 TPOT = 0.0
      DO 1300 I=1,LL
      NNONOC(I)=0
```

```
      NNTWOC(I)=0
      NNTHOC(I)=0
      NNFOOC(I)=0
      DO 1400 N=1,NUMNN
      GO TO (1410,1420,1430,1440),N
 1410 DO 1415 J=1,12
      K=NBRONE(J,I)
      NNONOC(I)=NNONOC(I)+NOCC(K)
 1415 CONTINUE
      GO TO 1400
 1420 DO 1425 J=1,6
      K=NBRTWO(J,I)
      NNTWOC(I)=NNTWOC(I)+NOCC(K)
 1425 CONTINUE
      GO TO 1400
 1430 DO 1435 J=1,24
      K=NBRTHR(J,I)
      NNTHOC(I)=NNTHOC(I)+NOCC(K)
 1435 CONTINUE
      GO TO 1400
 1440 DO 1445 J=1,12
      K=NBRFOR(J,I)
      NNFOOC(I)=NNFOOC(I)+NOCC(K)
 1445 CONTINUE
      GO TO 1400
 1400 CONTINUE
      DO 1301 N=1,NUMNN
      GO TO (1310,1320,1330,1340),N
 1310. PE(I)=PFIV*NNONOC(I)*PENN(1)
      GO TO 1301
 1320 PE(I)=PE(I)+PFIV*NNTWOC(I)*PENN(2)
      GO TO 1301
 1330 PE(I)=PE(I)+PFIV*NNTHOC(I)*PENN(3)
      GO TO 1301
 1340 PE(I)=PE(I)+PFIV*NNFOOC(I)*PENN(4)
      GO TO 1301
 1301 CONTINUE
      TPOT=TPOT+PE(I)
 1300 CONTINUE
 1201 ICOUNT=ICOUNT+1
      IF(ICOUNT) 1299,2900,1299
C
C     HERE'S WHERE THE ACTION IS - THIS IS THE PROBABILITY AND TRANSITION SEGMENT
C
 1299 DO 2000 I=1,LL
      IF(NOCC(I)) 2000,2000,2002
C
C     COMPUTE CURRENT NNONOC(I)
```

```
C
 2002 NNONOC(I)=0
      DO 2005 J=1,12
      K=NBRONE(J,I)
      NNONOC(I)=NNONOC(I)+NOCC(K)
 2005 CONTINUE
      IF(12-NNONOC(I)) 2000,2000,2010
 2010 DO 2050 J=1,12
      K=NBRONE(J,I)
      IF(K-IVAC) 2060,2075,2060
 2060 IF(NOCC(K)) 2075,2065,2075
C
C     SITE IS ACCESSIBLE - COMPUTE "DE"
C
 2065 DE(J,I)=PE(K)-PE(I)
      GO TO 2050
C
C     SITE IS OCCUPIED OR INACCESSIBLE - GIVE IT AN ARBITRARILY LARGE "DE"
C
 2075 DE(J,I)=1000.0
 2050 CONTINUE
C
C     INITIALIZE NEGATIVE "DE" INDICES, JLOE(J), TO ZERO
C
      DO 2100 J=1,12
      JLOE(J)=0
 2100 CONTINUE
      KOUNT1=0
 2140 DO 2150 J=1,12
      IF(DE(J,I)) 2160,2160,2150
 2160 KOUNT1=KOUNT1+1
      JLOE(KOUNT1)=J
 2150 CONTINUE
C
C     TEST FOR THE NUMBER OF DE'S .LE. ZERO
C
      IF(KOUNT1-1) 2400,2200,2300
C
C     CASE I: EXACTLY ONE DE .LE. ZERO - CARRY OUT THE TRANSITION
C
 2200 K=NBRONE(JLOE(1),I)
      NOCC(K)=1
      NOCC(I)=0
C
C     SINCE WE'VE MADE A TRANSITION, ADJUST THE POTENTIALS
C
C     "KNEW" (= PRESENT "K") IS THE SITE TO WHICH WE'VE JUST JUMPED
C     "I" IS THE SITE FROM WHICH WE'VE JUMPED, IE, THE SITE WE JUST VACATED
```

```
C
 2240 KNEW=K
      DO 2260 N=1,NUMNN
      GO TO (2241,2245,2249,2253), N
 2241 DO 2242 J=1,12
      K=NBRONE(J,I)
      PE(K)=PE(K)-PFIV*PENN(1)
 2242 CONTINUE
      GO TO 2260
 2245 DO 2246 J=1,6
      K=NBRTWO(J,I)
      PE(K)=PE(K)-PFIV*PENN(2)
 2246 CONTINUE
      GO TO 2260
 2249 DO 2250 J=1,24
      K=NBRTHR(J,I)
      PE(K)=PE(K)-PFIV*PENN(3)
 2250 CONTINUE
      GO TO 2260
 2253 DO 2254 J=1,12
      K=NBRFOR(J,I)
      PE(K)=PE(K)-PFIV*PENN(4)
 2254 CONTINUE
      GO TO 2260
 2260 CONTINUE
      DO 2280 N=1,NUMNN
      GO TO (2261,2265,2269,2273), N
 2261 DO 2262 J=1,12
      K=NBRONE(J,KNEW)
      PE(K)=PE(K)+PFIV*PENN(1)
 2262 CONTINUE
      GO TO 2280
 2265 DO 2266 J=1,6
      K=NBRTWO(J,KNEW)
      PE(K)=PE(K)+PFIV*PENN(2)
 2266 CONTINUE
      GO TO 2280
 2269 DO 2270 J=1,24
      K=NBRTHR(J,KNEW)
      PE(K)=PE(K)+PFIV*PENN(3)
 2270 CONTINUE
      GO TO 2280
 2273 DO 2274 J=1,12
      K=NBRFOR(J,KNEW)
      PE(K)=PE(K)+PFIV*PENN(4)
 2274 CONTINUE
      GO TO 2280
 2280 CONTINUE
```

```
      GO TO 2000
C
C     CASE II: SEVERAL DE .LE. ZERO - FIND LOWEST DE AND CARRY OUT THE
C     TRANSITION (IF LOWEST DE IS NOT UNIQUE, DECIDE SITE STOCHASTICALLY)
C
 2300 DO 2310 J=3,12
      IF(JLOE(J)) 2310,2315,2310
 2310 CONTINUE
 2315 IF(J-12) 2325,2320,2320
C
C     NO NN1'S: SHOULD BE IMPOSSIBLE, BUT WE WANT TO KNOW IF IT'S SO - PRINT MSG
C
 2320 WRITE (6,9220) I,J,IHNO1
 2325 JMAX=J-1
      EMIN=-DE(JLOE(1),I)
      KOUNT2=1
      MULT(1)=JLOE(1)
      DO 2350 J=2,JMAX
      IF(EMIN+DE(JLOE(J),I)+EPSLON) 2332,2340,2328
 2328 IF(EMIN+DE(JLOE(J),I)-EPSLON) 2340,2340,2350
 2332 IF(EMIN+DE(JLOE(J),I)-EPSLON) 2345,2340,2340
C
C     CASE II-A: ABS(EMIN) APPROXIMATELY .EQ. ABS(DE(JLOE(J),I)); DEGENERACY
C     IN ENERGY
C
 2340 KOUNT2=KOUNT2+1
      MULT(KOUNT2)=JLOE(J)
      GO TO 2350
C
C     CASE II-B: ABS(EMIN).LT.ABS(DE(JLOE(J),I)); DEFINE NEW EMIN
C
 2345 EMIN=-DE(JLOE(J),I)
      KOUNT2=1
      MULT(1)=JLOE(J)
      GO TO 2350
C
C     CASE II-C: ABS(EMIN).GT.ABS(DE(JLOE(J),I)); NO CHANGES REQUIRED
C
 2350 CONTINUE
      GO TO (2368,2353),KOUNT2
C
 2353 IRAN=IRAN*KRAN
      RANDOM=0.5+FLOAT(IRAN)*2.328306E-10
C
      KOUNTM=KOUNT2-1
      DO 2360 ITEST=1,KOUNTM
      IF(RANDOM-FTEST(ITEST,KOUNT2)) 2365,2365,2360
 2365 NA=ITEST
```

```
      GO TO 2370
 2360 CONTINUE
      NA=KOUNT2
      GO TO 2370
 2368 NA=1
C
C     CARRY OUT THE TRANSITION AND ADJUST THE POTENTIALS
C
 2370 K=NBRONE(MULT(NA),I)
      NOCC(K)=1
      NOCC(I)=0
      GO TO  2240
C
C     CASE III: ALL 12 DE .GT. ZERO - FIND LOWEST DE, DECIDE JUMP/NO JUMP
C     STOCHASTICALLY; IF NO JUMP FOR LOWEST DE, DO WE JUMP FOR NEXT LOWEST?
C     IF NOT, DO WE JUMP FOR NEXT LOWEST? ETC, ETC, ETC
C
 2400 EMIN=DE(1,I)
      KOUNT3=1
      MULT(1)=1
      DO 2420 J=2,12
      IF(EMIN-DE(J,I)+EPSLON) 2408,2410,2403
 2403 IF(EMIN-DE(J,I)-EPSLON) 2410,2410,2405
 2408 IF(EMIN-DE(J,I)-EPSLON) 2420,2410,2410
C
C     CASE III-A: EMIN.GT.DE(J,I); DEFINE NEW EMIN
C
 2405 EMIN=DE(J,I)
      KOUNT3=1
      MULT(1)=J
      GO TO 2420
C
C     CASE III-B: EMIN APPROXIMATELY .EQ. DE(J,I); DEGENERACY IN ENERGY
C
 2410 KOUNT3=KOUNT3+1
      MULT(KOUNT3)=J
C
C     CASE III-C: EMIN .LT. DE(J,I); NO CHANGES REQUIRED
C
 2420 CONTINUE
C
C     CHECK TO SEE IF EMIN CORRESPONDS TO FILLED OR INACCESSIBLE SITE
C
      IF(EMIN-900.0) 2430,2000,2000
C
C     AT LEAST ONE (MORE) SITE IS ACCESSIBLE-CALCULATE TRANSITION PROBABILITY
C
 2430 TPROB=EXP(-EMIN/TAU)
```

```
      KOUNT4=1
C
 2440 IRAN=IRAN*KRAN
      RANDOM=0.5+FLOAT(IRAN)*2.328306E-10
C
      IF(TPROB-RANDOM) 2445,2450,2450
 2445 KOUNT4=KOUNT4+1
      IF(KOUNT3-KOUNT4) 2455,2440,2440
C
C     TPROB.GE.RANDOM, SO CARRY OUT THE TRANSITION AND ADJUST THE POTENTIALS
C
 2450 K=NBRONE(MULT(KOUNT4),I)
      NOCC(I)=0
      NOCC(K)=1
      GO TO 2240
C
C     NO TRANSITION FOR PRESENT EMIN; FIND NEXT LOWER DE(J,I), AND ITS
C     MULTIPLICITY, AND TRY AGAIN
C
 2455 KOUNT5=1
      KOUNT6=1
      DO 2460 J=1,12
      IF(DE(J,I)-900.0) 2463,2460,2460
 2463 IF(DE(J,I)-EMIN-EPSLON) 2460,2460,2465
 2465 GO TO (2490,2470),KOUNT2
 2470 IF(EMINXT-DE(J,I)+EPSLON) 2478,2485,2473
 2473 IF(EMINXT-DE(J,I)-EPSLON) 2485,2485,2480
 2478 IF(EMINXT-DE(J,I)-EPSLON) 2460,2485,2485
C
C     CASE III-A: REDEFINE EMINXT
C
 2480 EMINXT=DE(J,I)
      KOUNT6=1
      MULT(1)=J
      GO TO 2460
C
C     CASE III-B: DEGENERACY IN ENERGY
C
 2485 KOUNT6=KOUNT6+1
      MULT(KOUNT6)=J
      GO TO 2460
C
C     WE'VE FOUND A CANDIDATE FOR EMINXT
C
 2490 EMINXT=DE(J,I)
      KOUNT5=2
      MULT(1)=J
C
```

```
C       CASE III-C: NO CHANGES REQUIRED
C
 2460 CONTINUE
      GO TO (2000,2495),KOUNT5
C
C       WE'VE FOUND AN ACCEPTABLE EMINXT, SO WE MAY STILL HAVE A TRANSITION;
C       REDEFINE PARAMETERS AND TRY TO JUMP AGAIN
C
 2495 EMIN=EMINXT
      KOUNT3=KOUNT6
      GO TO 2430
 2000 CONTINUE
 2900 CONTINUE
      DO 3010 K=1,IZ
      NUMSIT(K)=0
 3010 CONTINUE
      DO 3020 I=1,LL
      NPLANE=NZ(I)+1
      NUMSIT(NPLANE)=NUMSIT(NPLANE)+NOCC(I)
 3020 CONTINUE
      IF (ICOUNT) 3028,3021,3028
 3021 NUMATM=0
      DO 3025 K=1,IZ
      NUMATM=NUMATM+NUMSIT(K)
 3025 CONTINUE
      WRITE (6,9500)
      WRITE (6,9505) IHEAD
      WRITE (6,9510)
      WRITE (6,9630)
      WRITE (6,9640) EXA,EXB
      WRITE (6,9650) FXA
      WRITE (6,9670) ROEA
      WRITE (6,9680) CP0,CP1,CP2,CP3
      WRITE (6,9690) CF0,CF1,CF2
      WRITE (6,9700) ROEB
      WRITE(6,9710) CGD1,CGD2,CGB1,CGB2
      WRITE (6,9720) CGF1,CGF2
      WRITE (6,9730) ROEC
      WRITE (6,9540) NUMNN
      WRITE (6,9740) (DIST(I),I=1,NUMNN)
      WRITE (6,9750) (PENN(I),I=1,NUMNN)
      WRITE (6,9620) KRAN,IRANI
      WRITE (6,9520) IX,IY,IZ
      WRITE (6,9530) LL,NUMATM
      WRITE (6,9550) EQTEMP, NUMRUN
      WRITE (6,9560) MSUM,MPIX
      WRITE (6,9760) MCRIT
      WRITE (6,9570)
```

```
 3028 IF (ICOUNT-MCRIT) 3029,3030,3030
 3029 IF(ICOUNT-(ICOUNT/MPIX)*MPIX) 3060,3030,3060
 3030 IF(IX-20) 3035,3200,3035
 3035 IF(IX-26) 3040,3300,3040
 3040 IF(IX-10) 3500,3400,3500
C
C     THIS SEGMENT PRINTS OUT PICTURES OF THE MICROSTATE FOR IX = 20
C
 3200 WRITE(6,9580) ICOUNT
      DO 3250  K=1,IZ,2
      WRITE (6,9340)
      WRITE (6,9346)
      KM1=K-1
      KP1=K+1
      DO 3220 J=1,IY,2
      JJ=IY-J
      JJM=JJ-1
      JP1=J+1
      KA=K*ISZ
      KB=KA-J*ISX+1
      KC=KA-JP1*ISX+1
      KD=KB+ISXM
      KE=KC+ISXM
      KF=KP1*ISZ
      KG=KF-J*ISX+1
      KH=KF-JP1*ISX+1
      KI=KG+ISXM
      KJ=KH+ISXM
      WRITE (6,9321) JJ,(NOCC(I),I=KB,KD),JJ,(NOCC(I),I=KG,KI)
 3220 WRITE (6,9331) JJM,(NOCC(I),I=KC,KE),JJM,(NOCC(I),I=KH,KJ)
      WRITE (6,9351)
      WRITE (6,9356)
      WRITE(6,9360) KM1,K
 3250 WRITE (6,9370) NUMSIT(K),NUMSIT(KP1)
      WRITE (6,9625) IRAN,RANDOM
      IF (ICOUNT-NUMRUN) 7900,8000,8000
C
C     THIS SEGMENT PRINTS OUT PICTURES OF THE MICROSTATE FOR IX = 26
C
 3300 WRITE(6,9580) ICOUNT
      DO 3350  K=1,IZ,2
      WRITE (6,9340)
      WRITE (6,9347)
      KM1=K-1
      KP1=K+1
      DO 3320 J=1,IY,2
      JJ=IY-J
      JJM=JJ-1
```

```
      JP1=J+1
      KA=K*ISZ
      KB=KA-J*ISX+1
      KC=KA-JP1*ISX+1
      KD=KB+ISXM
      KE=KC+ISXM
      KF=KP1*ISZ
      KG=KF-J*ISX+1
      KH=KF-JP1*ISX+1
      KI=KG+ISXM
      KJ=KH+ISXM
      WRITE (6,9322)  JJ,(NOCC(I),I=KB,KD),JJ,(NOCC(I),I=KG,KI)
 3320 WRITE (6,9332)  JJM,(NOCC(I),I=KC,KE),JJM,(NOCC(I),I=KH,KJ)
      WRITE (6,9352)
      WRITE (6,9357)
      WRITE(6,9360) KM1,K
 3350 WRITE (6,9370)  NUMSIT(K),NUMSIT(KP1)
      WRITE (6,9625)  IRAN,RANDOM
      IF (ICOUNT-NUMRUN) 7900,8000,8000
C
C     THIS SEGMENT PRINTS OUT PICTURES OF THE MICROSTATE FOR IX = 10
C
 3400 WRITE(6,9580)  ICOUNT
      DO 3450  K=1,IZ,2
      WRITE (6,9340)
      WRITE (6,9345)
      KM1=K-1
      KP1=K+1
      DO 3420 J=1,IY,2
      JJ=IY-J
      JJM=JJ-1
      JP1=J+1
      KA=K*ISZ
      KB=KA-J*ISX+1
      KC=KA-JP1*ISX+1
      KD=KB+ISXM
      KE=KC+ISXM
      KF=KP1*ISZ
      KG=KF-J*ISX+1
      KH=KF-JP1*ISX+1
      KI=KG+ISXM
      KJ=KH+ISXM
      WRITE (6,9320)  JJ,(NOCC(I),I=KB,KD),JJ,(NOCC(I),I=KG,KI)
 3420 WRITE (6,9330)  JJM,(NOCC(I),I=KC,KE),JJM,(NOCC(I),I=KH,KJ)
      WRITE (6,9350)
      WRITE (6,9355)
      WRITE(6,9360) KM1,K
 3450 WRITE (6,9370)  NUMSIT(K),NUMSIT(KP1)
```

```
      WRITE (6,9625) IRAN,RANDOM
      IF (ICOUNT-NUMRUN) 7900,8000,8000
C
C     THIS SEGMENT PRINTS OUT PICTURES OF THE MICROSTATE
C     FOR VALUES OF IX OTHER THAN 10, 20 AND 26
C
 3500 WRITE (6,9580) ICOUNT
      DO 3580 K=1,IZ,2
      WRITE (6,9341)
      WRITE (6,9348)
      KM1=K-1
      KP1=K+1
      DO 3520 J=1,IY,2
      JJ=IY-J
      JJM=JJ-1
      JP1=J+1
      KA=K*ISZ
      KB=KA-J*ISX+1
      KC=KA-JP1*ISX+1
      KD=KB+ISXM
      KE=KC+ISXM
      WRITE (6,9323) JJ,(NOCC(I),I=KB,KD)
 3520 WRITE (6,9333) JJM,(NOCC(I),I=KC,KE)
      WRITE (6,9353)
      WRITE (6,9358)
      WRITE(6,9361) KM1
      WRITE (6,9371) NUMSIT(K)
      WRITE (6,9341)
      WRITE (6,9348)
      DO 3540 J=1,IY,2
      JJ=IY-J
      JJM=JJ-1
      JP1=J+1
      KF=KP1*ISZ
      KG=KF-J*ISX+1
      KH=KF-JP1*ISX+1
      KI=KG+ISXM
      KJ=KH+ISXM
      WRITE (6,9333) JJ,(NOCC(I),I=KH,KJ)
 3540 WRITE (6,9323) JJM,(NOCC(I),I=KG,KI)
      WRITE (6,9353)
      WRITE (6,9358)
      WRITE(6,9361) K
 3580 WRITE (6,9371) NUMSIT(KP1)
      WRITE (6,9625) IRAN,RANDOM
      IF (ICOUNT-NUMRUN) 7900,8000,8000
C
C     THIS "DO" LOOP (THRU '3070') PRINTS OUT "SUMMARY INFO" OF THE MICROSTATE
```

```
C
 3060 IF (ICOUNT-(ICOUNT/MSUM)*MSUM) 7900,3061,7900
 3061 WRITE (6,9590) ICOUNT
      WRITE (6,9600)
      DO 3070 K=1,IZ,4
      KM1=K-1
      KP1=K+1
      KP2=K+2
      KP3=K+3
      IF (KP2-IZ) 3063,3063,3067
 3063 WRITE (6,9610) KM1,NUMSIT(K),K,NUMSIT(KP1),KP1,
     1NUMSIT(KP2),KP2,NUMSIT(KP3)
      GO TO 3070
 3067 WRITE (6,9610) KM1,NUMSIT(K),K,NUMSIT(KP1)
 3070 CONTINUE
      IF (ICOUNT-NUMRUN) 7900,8000,8000
 7900 IF(ICOUNT-(ICOUNT/3)*3) 1201,1202,1201
 8000 IF (NCRYST) 9,8100,9
 8100 STOP
      END


      SUBROUTINE CROSYM
C     SIMULTANEOUS EQUATION SOLVER                                      JOB,650
C     LOCKHEED MISSLES AND SPACE COMPANY, SUNNYVALE, CALIFORNIA         JOB,651
C     SOLVE M SIMULTANEOUS EQUATIONS BY THE METHOD OF CROUT             JOB,652
      COMMON/COMA/ A(4,5),MCRO
      M=MCRO
      N=M+1                                                             JOB,654
      I1=1                                                              JOB,655
  100 I3=I1                                                             JOB,656
      SUM=ABS(A(I1,I1))                                                 JOB,657
      DO 120 I=I1,M                                                     JOB,658
      IF(SUM-ABS(A(I,I1))) 110,120,120                                  JOB,659
  110 I3=I                                                              JOB,660
      SUM=ABS(A(I,I1))                                                  JOB,661
  120 CONTINUE                                                          JOB,662
      IF(I3-I1) 130,150,130                                             JOB,663
  130 DO 140 J=1,N                                                      JOB,664
      SUM=-A(I1,J)                                                      JOB,665
      A(I1,J)=A(I3,J)                                                   JOB,666
  140 A(I3,J)=SUM                                                       JOB,667
  150 I3=I1+1                                                           JOB,668
      DO 160 I=I3,M                                                     JOB,669
  160 A(I,I1)=A(I,I1)/A(I1,I1)                                          JOB,670
  170 J2=I1-1                                                           JOB,671
      I3=I1+1                                                           JOB,672
      IF(J2) 180,200,180                                                JOB,673
```

```
  180 DO 190 J=I3,N
      DO 190 I=1,J2
  190 A(I1,J)=A(I1,J)-A(I1,I)*A(I,J)
      IF(I1-M) 200,220,200
  200 J2=I1
      I1=I1+1
      DO 210 I=I1,M
      DO 210 J=1,J2
  210 A(I,I1)=A(I,I1)-A(I,J)*A(J,I1)
      IF(I1-M) 100,170,100
  220 DO 240 I=1,M
      J2=M-I
      I3=J2+1
      A(I3,N)=A(I3,N)/A(I3,I3)
      IF(J2) 230,250,230
  230 DO 240 J=1,J2
  240 A(J,N)=A(J,N)-A(I3,N)*A(J,I3)
  250 RETURN
      END
//GO.FT06F001 DD SPACE=(CYL,(10,2),RLSE)
//GO.SYSIN DD *
 COPPER GB-2  63.54 GIBSON-2  10.02407  -9.19667  3.615
 ( ANDERMAN - NN(2) POTENTIAL )           0.83 1.10 2.40 0.48056 1.40465 2.627684
  NO NN1'S
INITIAL CONFIGURATION - PERFECT SURFACE WITH MONATOMIC STEP PARALLEL TO X-AXIS
   20   20    4  800
   10    1  450  500    0
   0.5        2
```

## TABLE I

### POTENTIAL FUNCTIONS AND COEFFICIENTS

$V_{\mathrm{Gibson\ II}}(R)$ = exp(EXA+EXB*R)

$V_{\mathrm{cubic}}(R)$ = R*(R*(R*CP3+CP2)+CP1)+CP0

$V_{\mathrm{trun\ Morse}}(R)$ = exp(CGD1+CGB1*R) - exp(CGD2+CGB2*R)

| Type Potential | Coefficients | Cutoff Distance (lattice units) |
|---|---|---|
| Gibson II | EXA = 10.02407<br>EXB = -9.19667 | $r_A$ = 0.83 |
| cubic | CP0 = 587.6182<br>CP1 = -1593.863<br>CP2 = 1450.286<br>CP3 = -442.2820 | $r_B$ = 1.10 |
| truncated Morse | CGD1 = 6.649136<br>CGD2 = 3.651313<br>CGB1 = -5.077808<br>CGB2 = -2.538904 | $r_C$ = 2.40 |

TABLE II

RESULTS OF A SINGLE ATOM JUMPING ON A PERFECT SURFACE

Note: L.U. = distance in lattice units

| Left | | | Right | | |
|---|---|---|---|---|---|
| 1 L.U. | 2 L.U. | 3 L.U. | 1 L.U. | 2 L.U. | 3 L.U. |
| 0.23 | 0.09 | 0.02 | 0.28 | 0.12 | 0.03 |

| Up | | | | | Down | | | |
|---|---|---|---|---|---|---|---|---|
| 1 L.U. | 2 L.U. | 3 L.U. | 4 L.U. | 5 L.U. | 1 L.U. | 2 L.U. | 3 L.U. | 4 L.U. |
| 0.12 | 0.08 | 0.03 | 0.01 | 0.01 | 0.40 | 0.02 | 0.00 | 0.00 |

| Vertical Motion Only | Lateral Motion Only | No Motion |
|---|---|---|
| 0.06 | 0.15 | 0.17 |

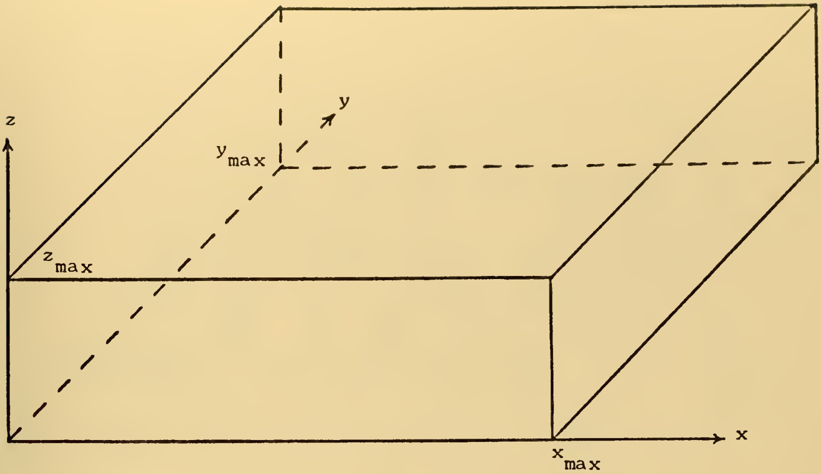


Figure 1. The Active Lattice Volume.

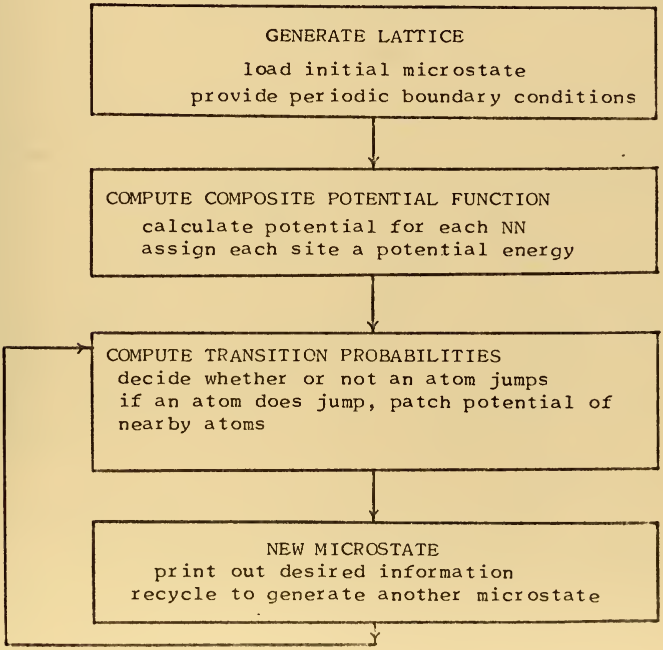

GENERATE LATTICE
load initial microstate
provide periodic boundary conditions
COMPUTE COMPOSITE POTENTIAL FUNCTION
calculate potential for each NN
assign each site a potential energy
COMPUTE TRANSITION PROBABILITIES
decide whether or not an atom jumps
if an atom does jump, patch potential of
nearby atoms
NEW MICROSTATE
print out desired information
recycle to generate another microstate

Figure 2. Overall Program Flow.

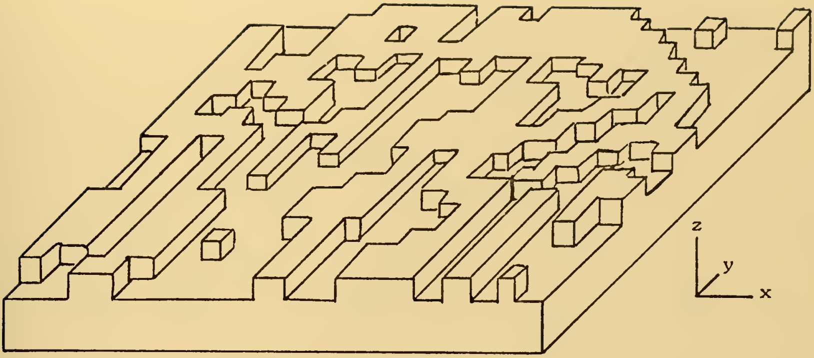
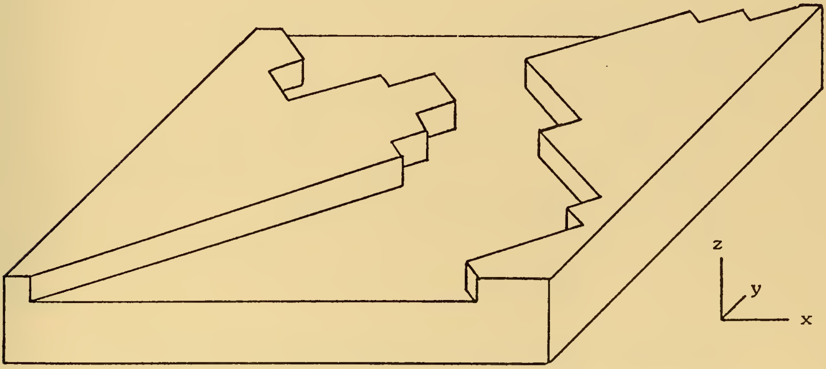


Figure 3. Perfect Surface Plus 99 Randomly Placed Atoms. (a) Initial microstate (lines bordering atoms and atom clusters are parallel and perpendicular to the ⟨100⟩ direction).



(b) Microstate 140 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction).

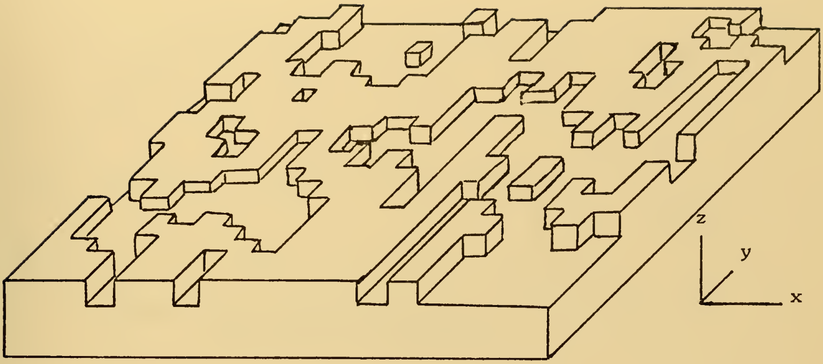
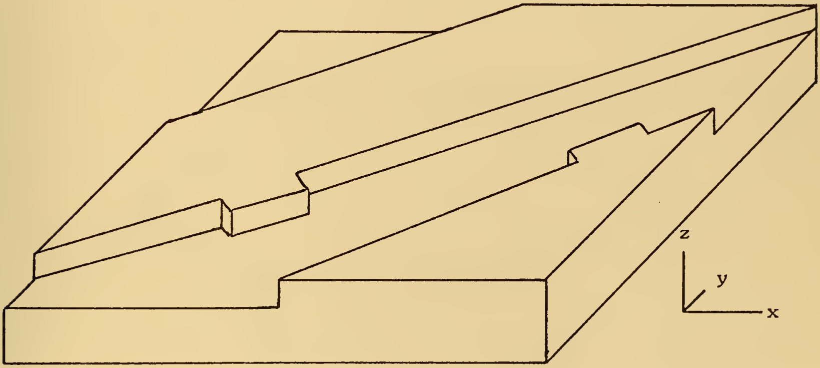


Figure 4. Perfect Surface Plus 104 Randomly Placed Atoms.
(a) Initial microstate (lines bordering atoms and atom clusters are parallel and perpendicular to the ⟨100⟩ direction).



(b) Microstate 140 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction).

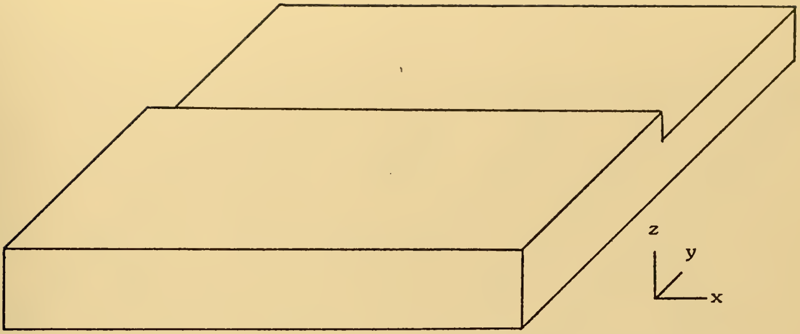


Figure 5. Perfect Surface Plus Half Plane Monatomic Step;
(a) initial microstate

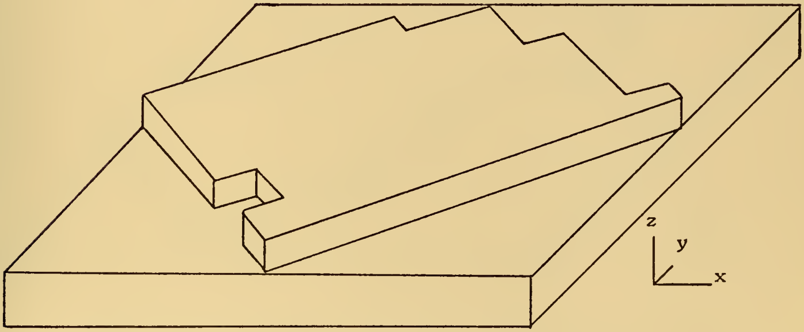


Figure 5b. Microstate 500 (lines bordering the atom cluster are parallel and perpendicular to the ⟨110⟩ direction)

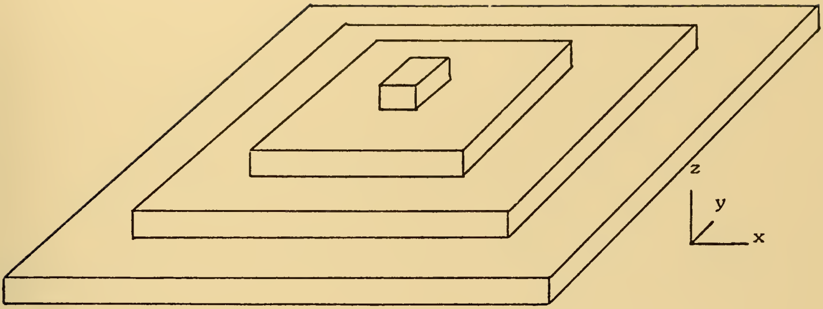


Figure 6. Pyramid With Steps Three Atomic Layers Wide; (a) initial microstate.

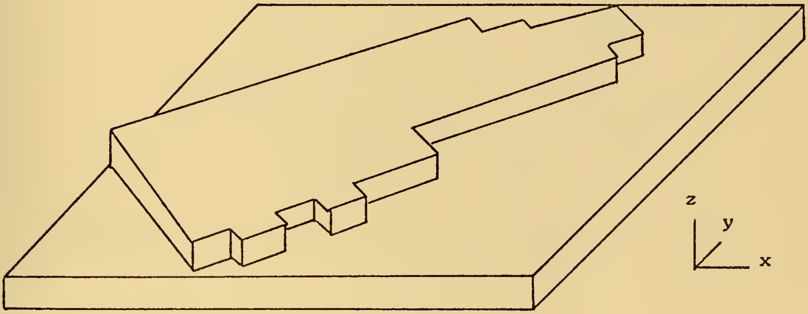


Figure 6b. Microstate 500 (lines bordering the atom cluster are parallel and perpendicular to the ⟨110⟩ direction).

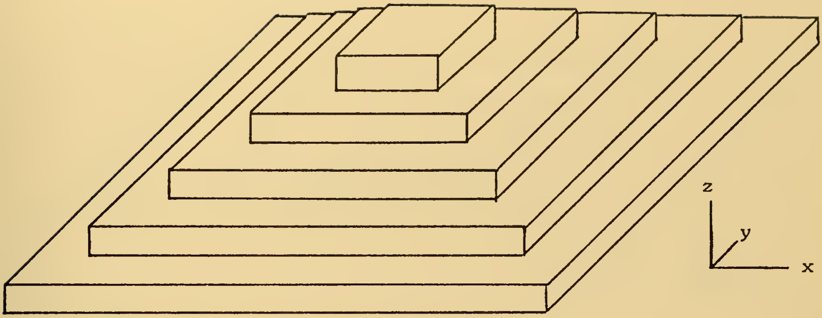
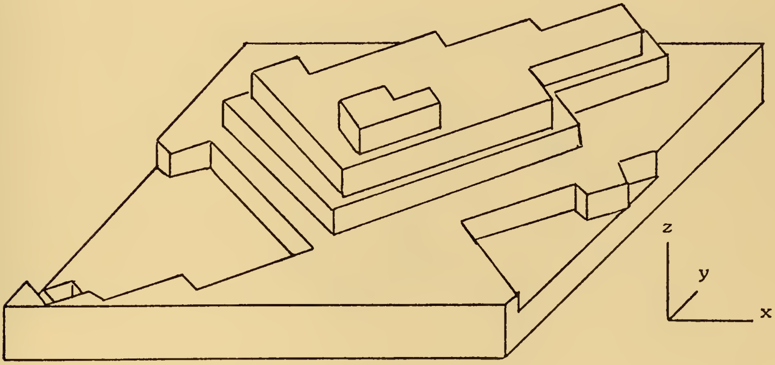


Figure 7. Pyramid With Steps Two Atomic Layers Wide.
(a) Initial microstate.



(b) Microstate 500 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction); the hole in the $z = 1$ plane is a result of the partial fill-in and reorientation of the initial grooves in the $z = 1$ plane; however, the vacancy in the $z = 0$ plane (lower left corner) is due to an atom jumping out of a perfect surface (the $z = 0$ plane) - this phenomenon was observed infrequently.

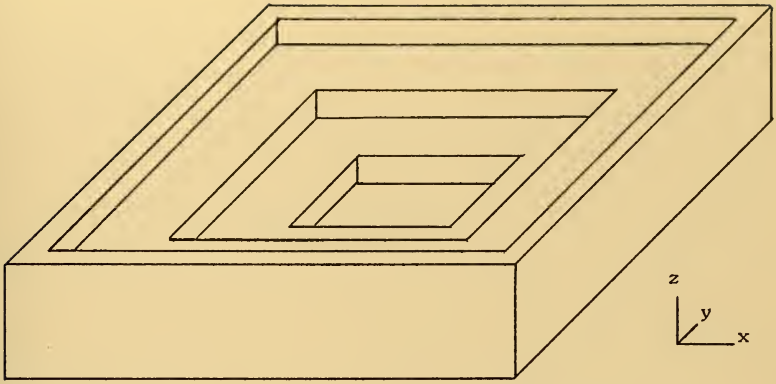


Figure 8. Pyramid Hole With Steps Three Atomic Layers Wide; (a) initial microstate.

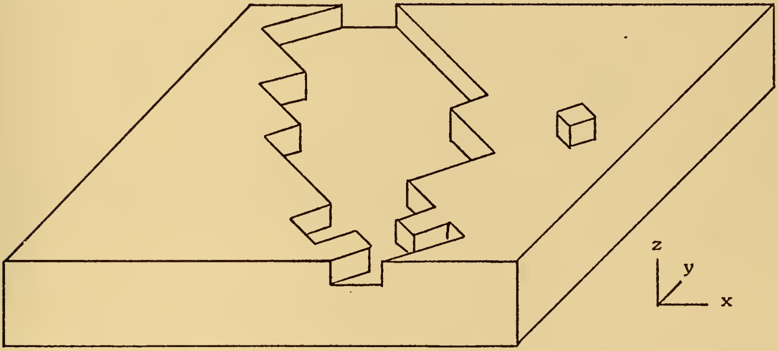


Figure 8b. Microstate 500 (lines bordering atoms and atom clusters are parallel and perpendicular to the ⟨110⟩ direction).

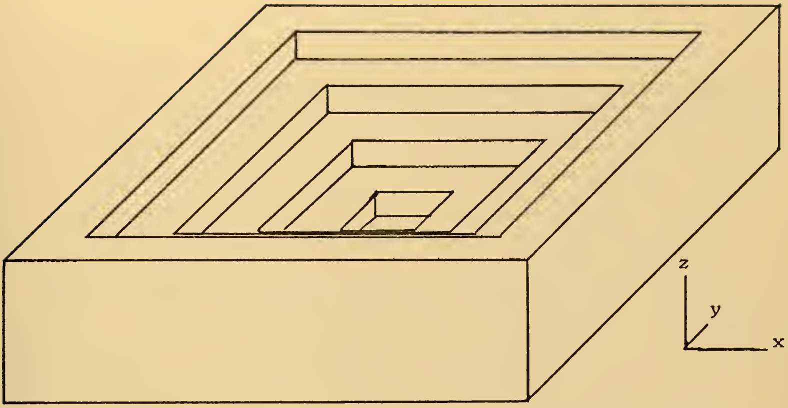
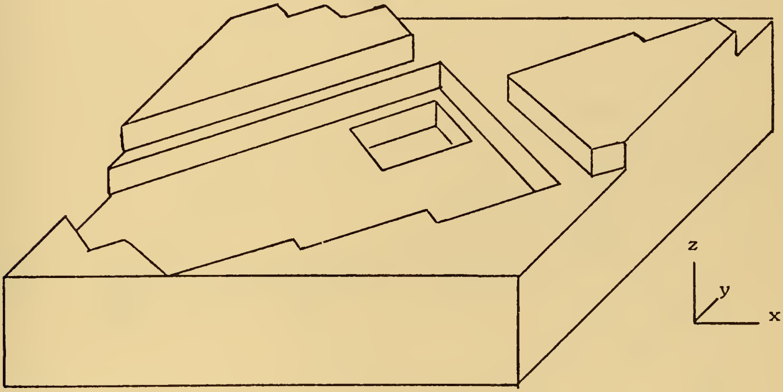


Figure 9. Pyramid Hole With Steps Two Atomic Layers Wide.
(a) Initial microstate.



(b) Microstate 500 (lines bordering clusters of atoms are parallel and perpendicular to the ⟨110⟩ direction); the holes appearing here are due to the partial fill-in and reorientation of the initial holes.

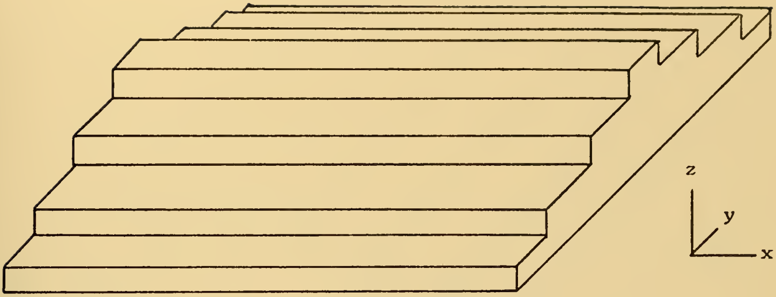
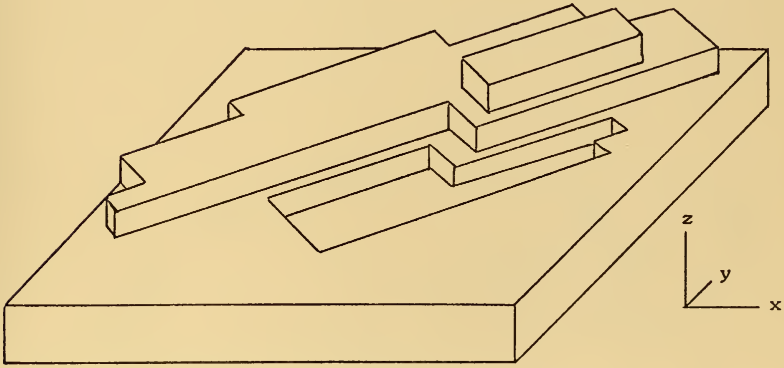


Figure 10. Ridge Parallel to X-Axis With Steps Three Atomic Layers Wide. (a) Initial microstate.



(b) Microstate 500 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction); the hole appearing in the $z = 1$ plane is due to the partial fill-in and reorientation of the initial groove in the $z = 1$ plane.

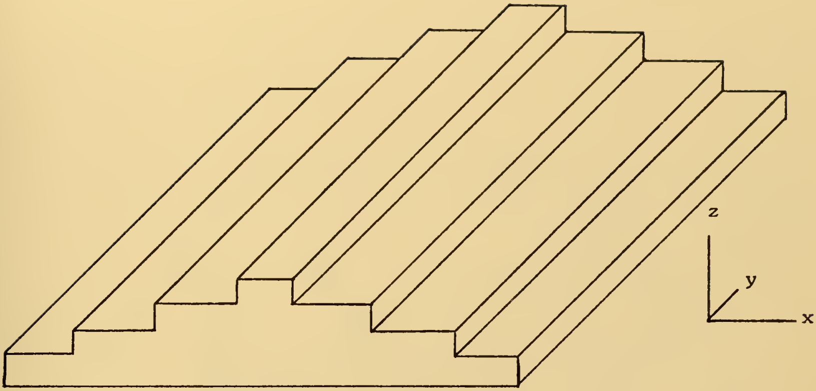
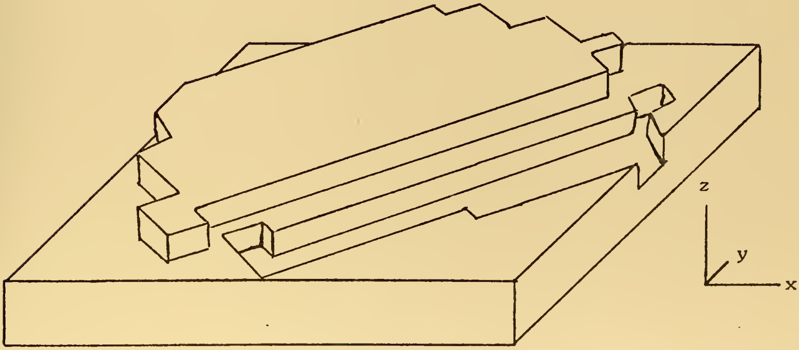
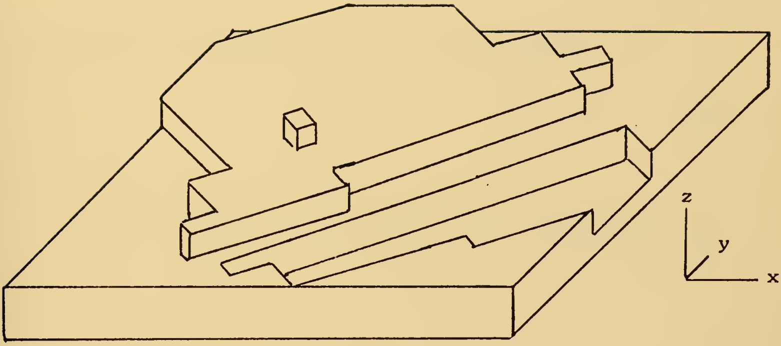


Figure 11. Ridge Parallel to Y-Axis With Steps Three Atomic Layers Wide. (a) Initial microstate.



(b) Microstate 500 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction); the hole appearing in the $z = 1$ plane is a result of the partial fill-in and reorientation of the initial groove in the $z = 1$ plane.



(c) Microstate 451, using a different random number seed then used to achieve the microstate depicted in Figure 11b (the lines bordering atoms and atom clusters are parallel and perpendicular to the ⟨110⟩ direction); the hole appearing in the z = 1 plane is a result of the partial fill-in and reorientation of the initial groove in the z = 1 plane.

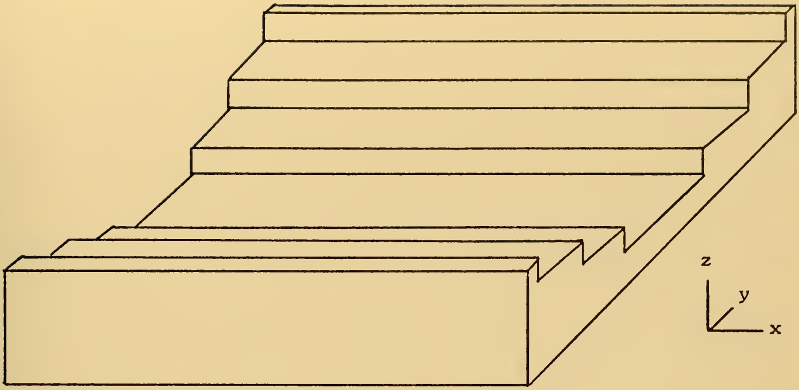


Figure 12. Valley Parallel to X-Axis With Steps Three Atomic Layers Wide; (a) initial microstate.

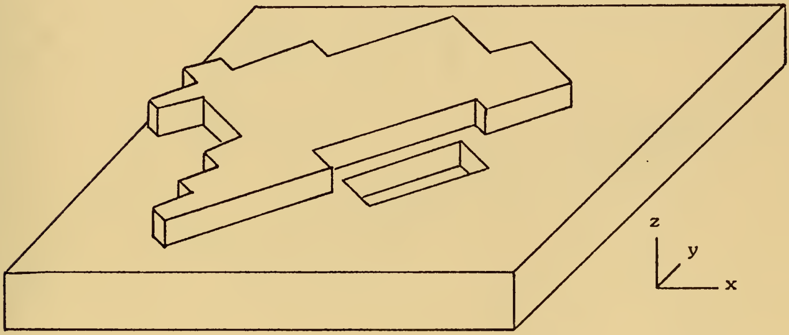


Figure 12b. Microstate 500 (lines bordering the atom clusters are parallel and perpendicular to the ⟨110⟩ direction). The hole appearing here is a result of the partial fill-in and reorientation of the initial groove in the z=1 plane.

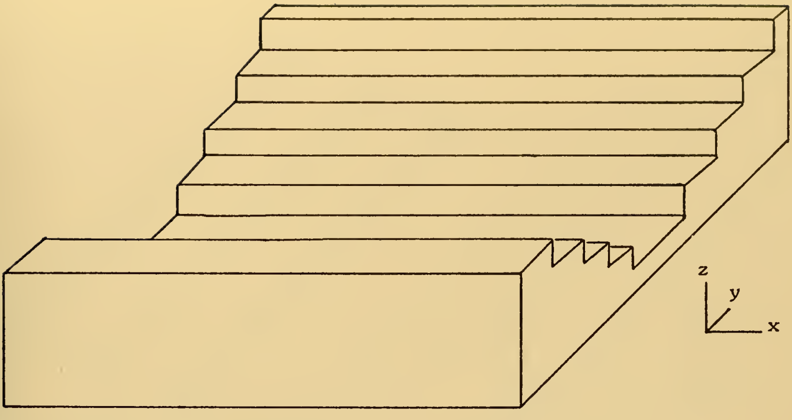


Figure 13. Valley Parallel to X-Axis With Steps Two Atomic Layers Wide; (a) initial microstate.

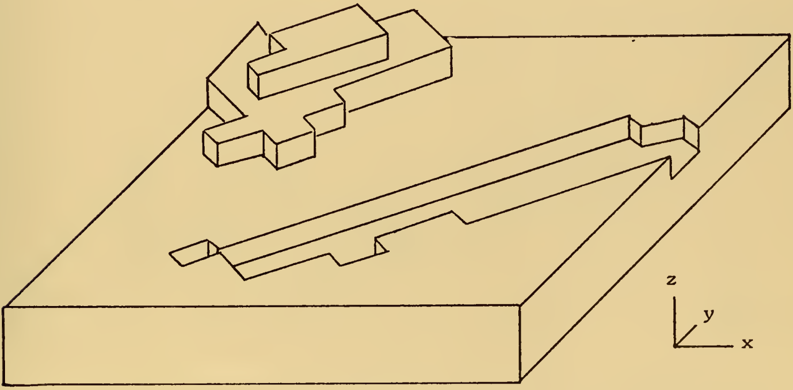


Figure 13b. Microstate 500 (lines bordering the atom clusters are parallel and perpendicular to the ⟨110⟩ direction). The hole appearing here is a result of the partial fill-in and reorientation of the initial groove in the z=2 plane.

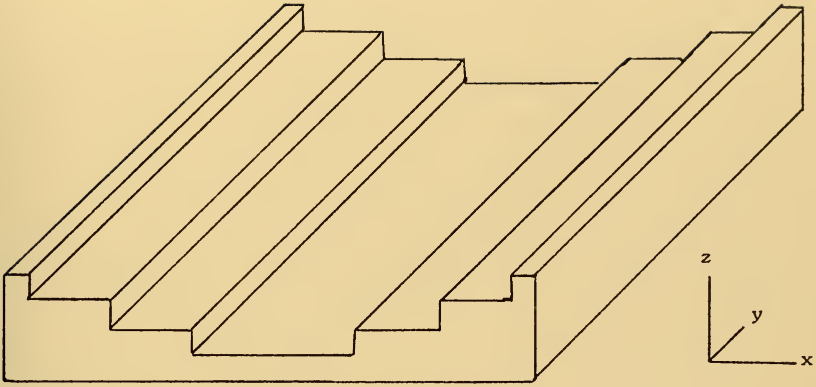
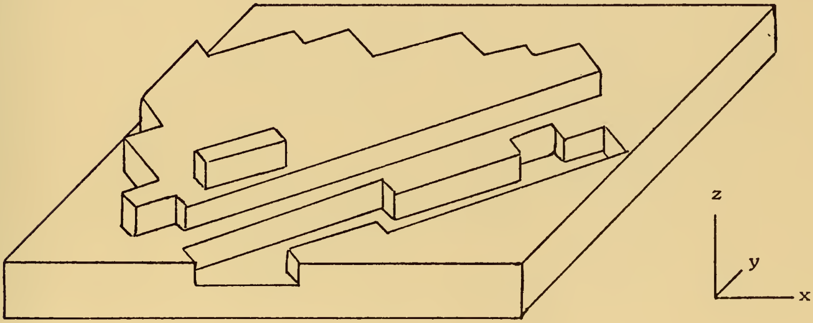


Figure 14. Valley Parallel to Y-Axis With Steps Three Atomic Layers Wide. (a) Initial microstate.



(b) Microstate 500 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction); the hole appearing in the z = 1 plane is a result of the partial fill-in and reorientation of the initial groove in the z = 1 plane.

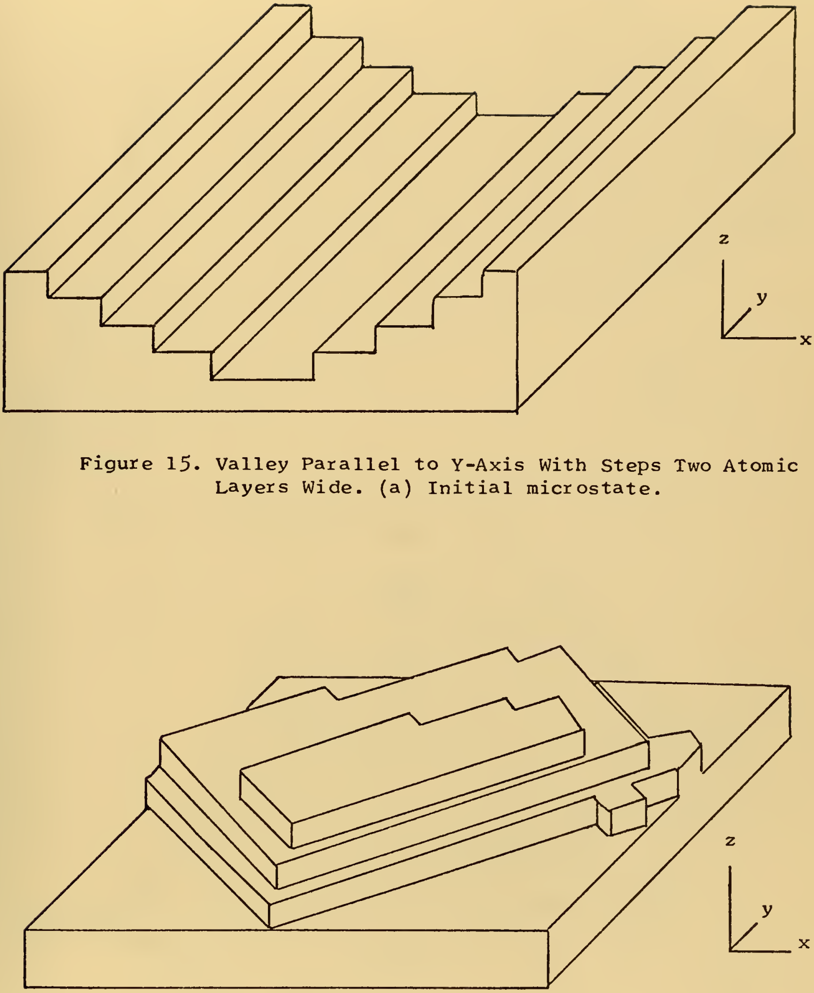


Figure 15. Valley Parallel to Y-Axis With Steps Two Atomic Layers Wide. (a) Initial microstate.

(b) Microstate 500 (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction).

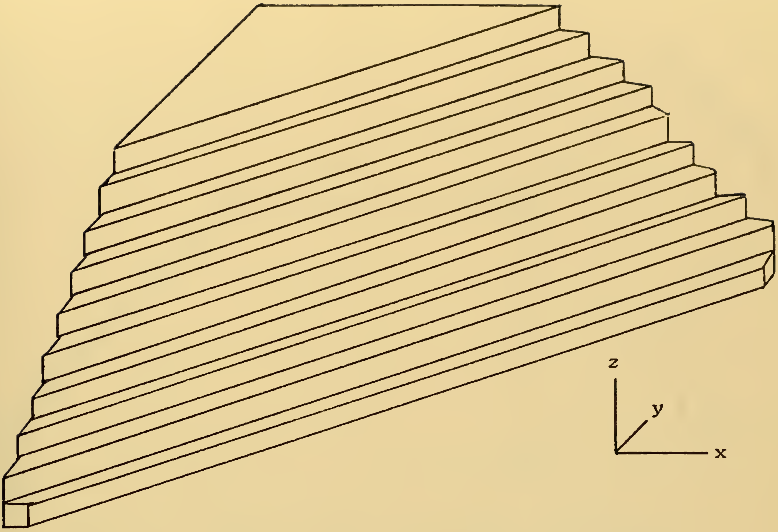
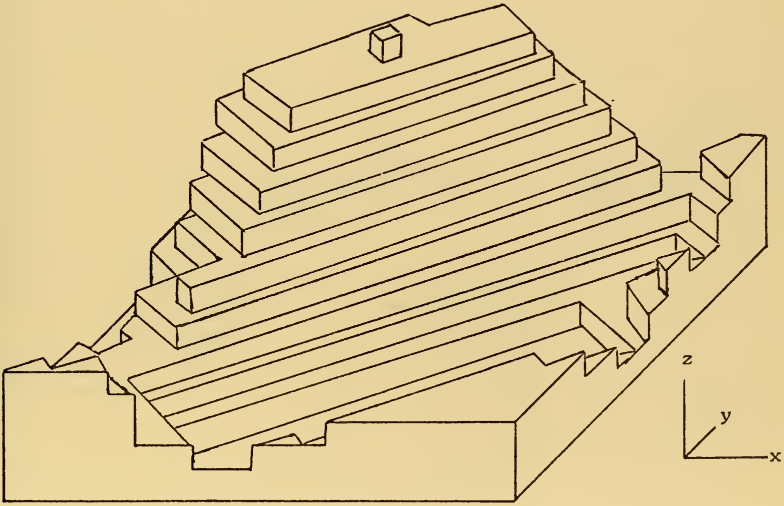


Figure 16. (1 -1 1) Plane, Truncated. (a) Initial microstate.



(b) Microstate 500.

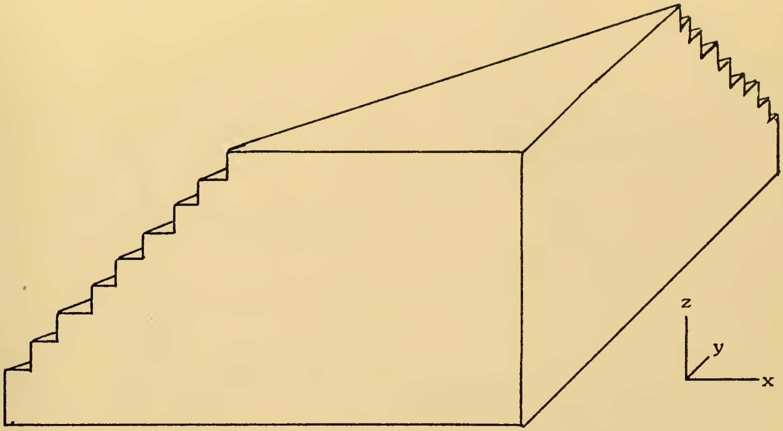
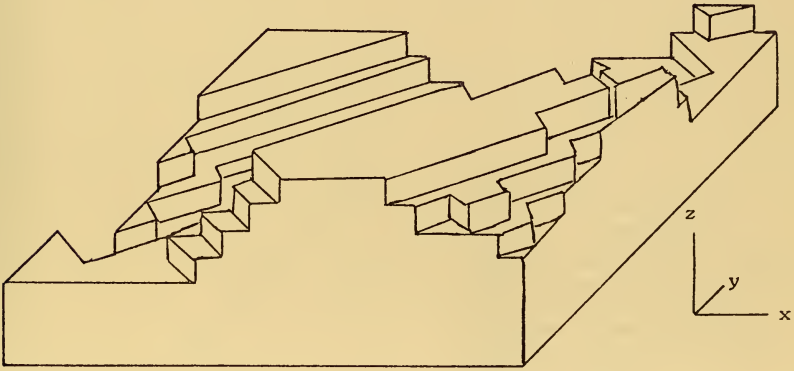


Figure 17. (-1 1 1) Plane, Truncated. (a) Initial microstate.



(b) Microstate 500.

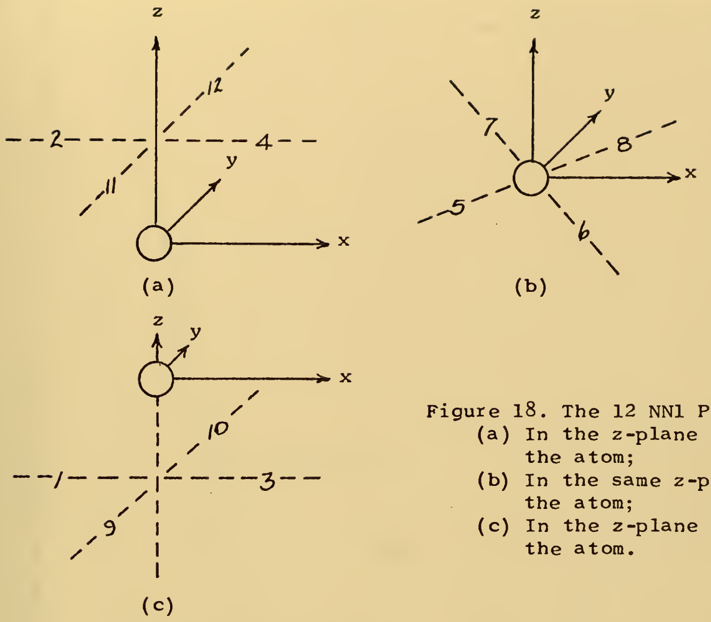


Figure 18. The 12 NN1 Positions.
(a) In the z-plane above the atom;
(b) In the same z-plane as the atom;
(c) In the z-plane below the atom.

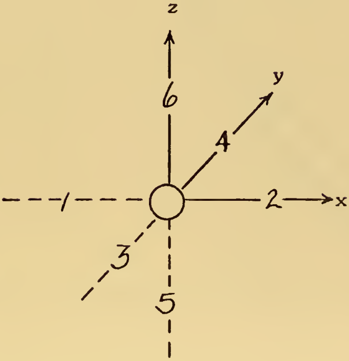


Figure 19. The Six NN2 Positions.

Sites at positions 1-4 are in the same z-plane as the atom; position 5 is two z-planes below the atom; position 6 is two z-planes above the atom.

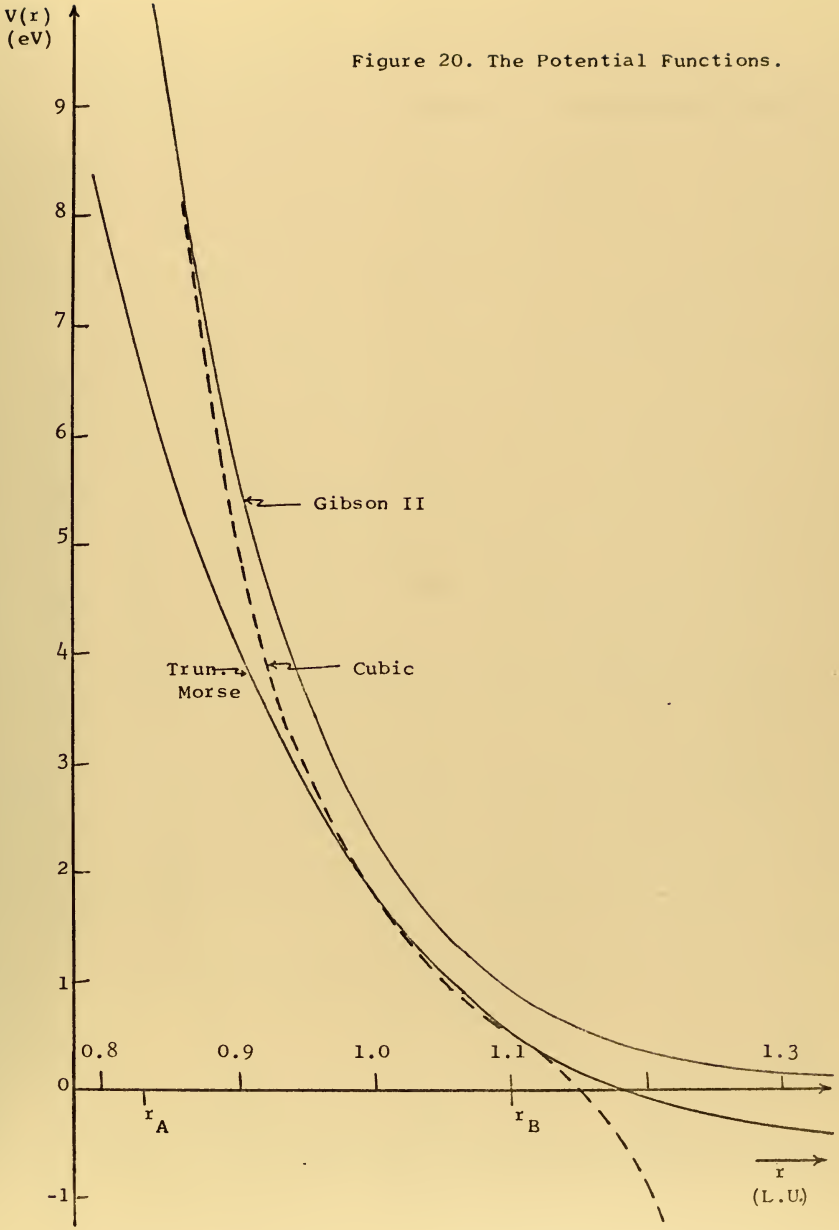


Figure 20. The Potential Functions.

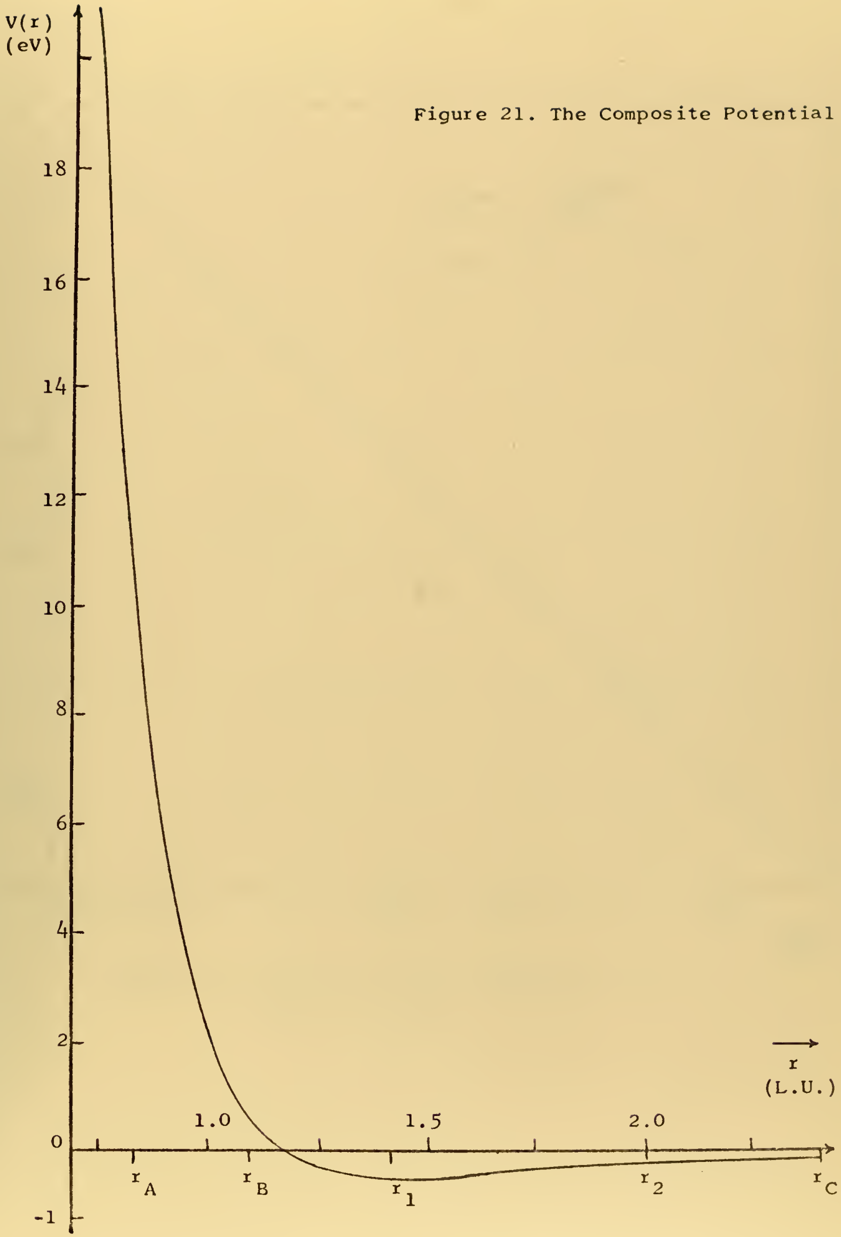


Figure 21. The Composite Potential

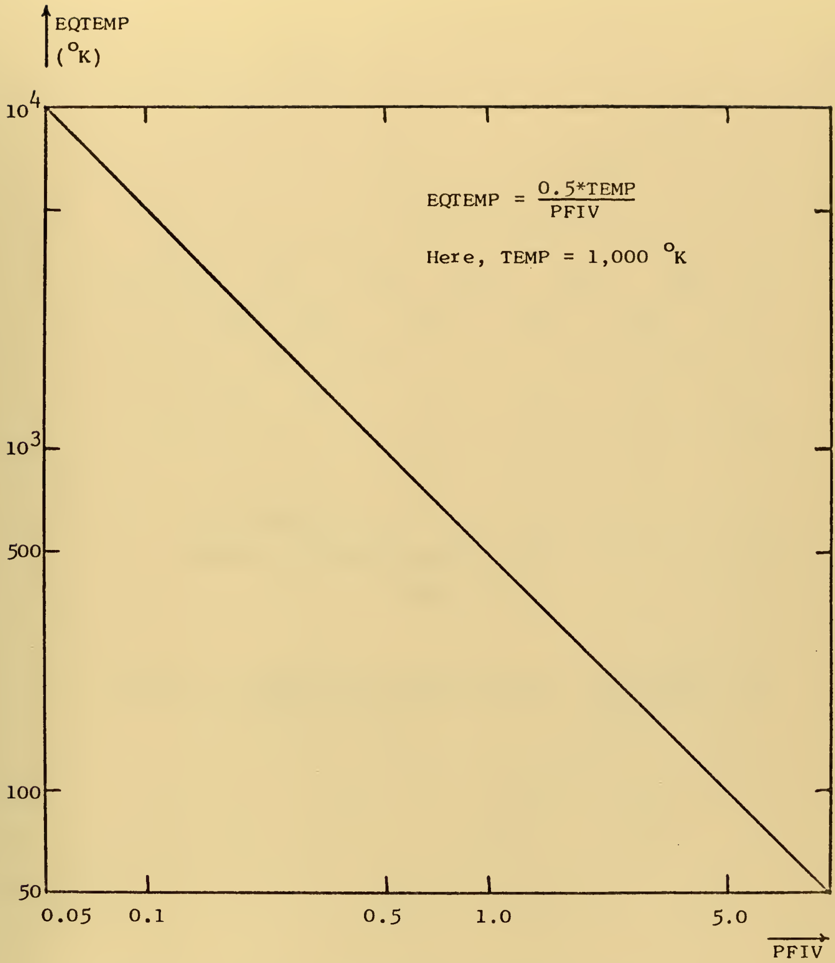
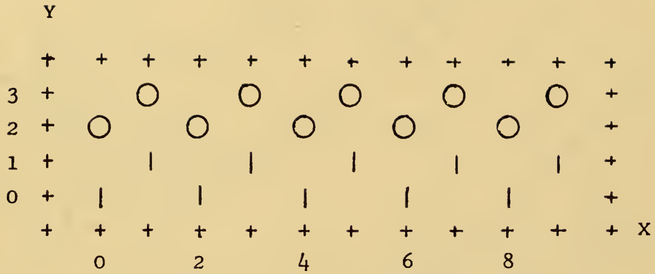


Figure 22. Equivalent Lattice Temperature as a Function of PFIV.



Z = 0 PLANE

NUMBER OF SITES OCCUPIED = 10

Figure 23. Sample Microstate Picture. Here, IX = 10 and IY = 4.

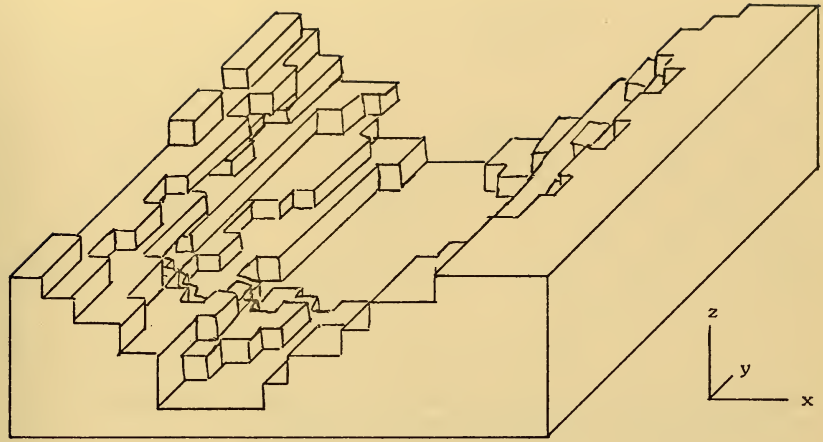
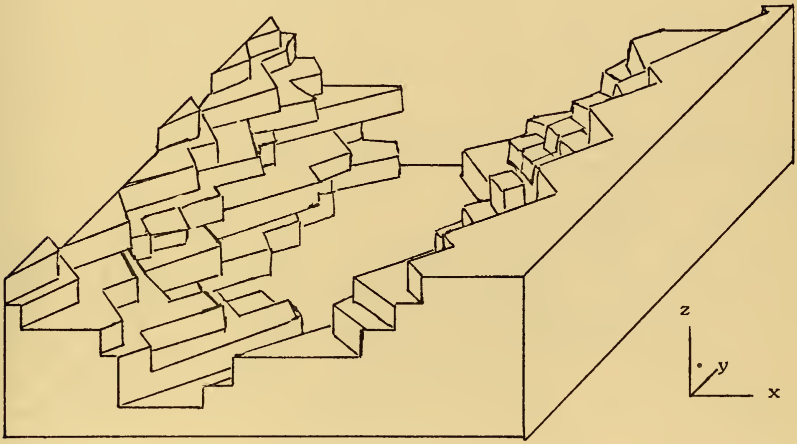


Figure 24. Microstate 310 of an Initially Perfect Surface at 10,000 °K. (a) Lines bordering atoms and clusters of atoms are parallel and perpendicular to the ⟨100⟩ direction.



(b) Lines bordering atoms and clusters of atoms are parallel and perpendicular to the ⟨110⟩ direction.

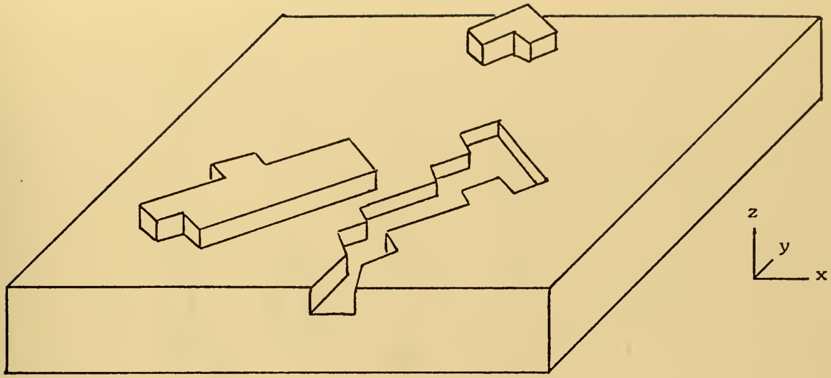


Figure 25. Microstate 180 of an Initially Perfect Surface at 2,000 °K (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction). Note that atoms have jumped out of the perfect surface and coalesced on the surface.

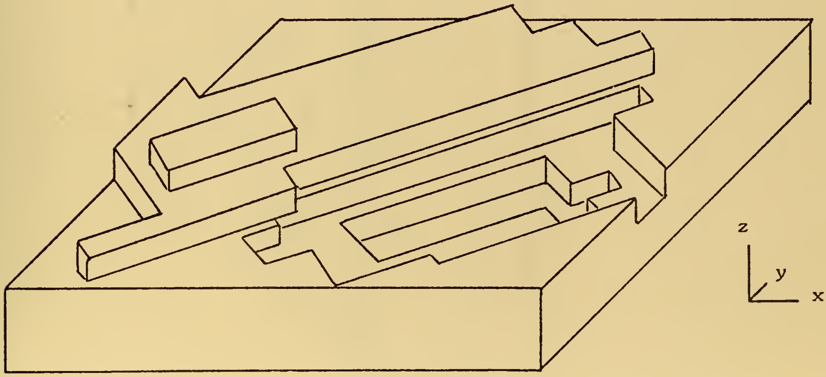


Figure 26. Microstate 190 of an Initially Perfect Surface at 2,584 °K (lines bordering atom clusters are parallel and perpendicular to the ⟨110⟩ direction). Note that a hole two atomic layers deep has been burroughed as atoms left the perfect surface and piled up near it.

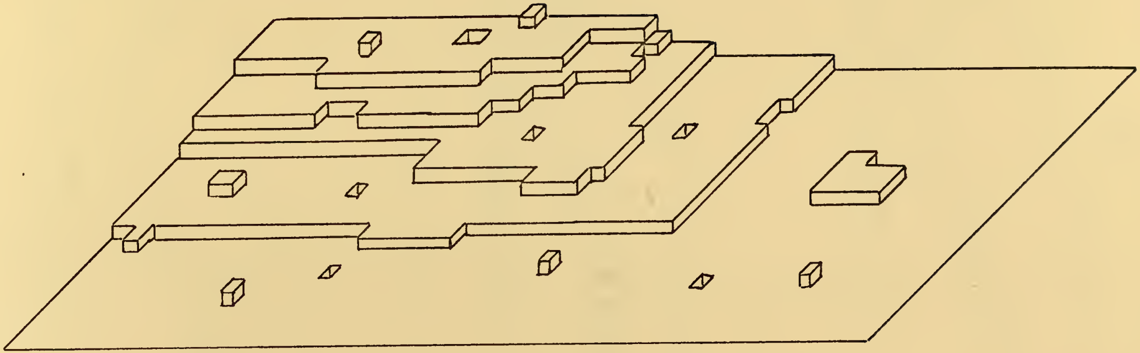


Figure 27. Harrison's Conception of a Surface Equilibrium Microstate.

Figure 28. Break Repair Sequence For Atom Aggregates in a Single Plane. The initial microstate was Figure 10a.

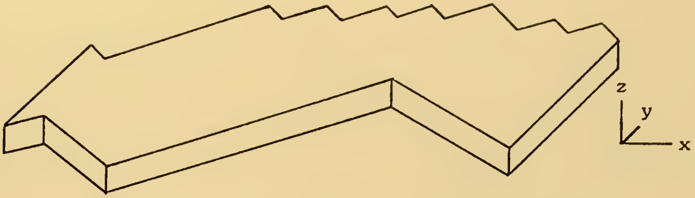


Figure 28a. Microstate 50, z = 2 plane (75 atoms).

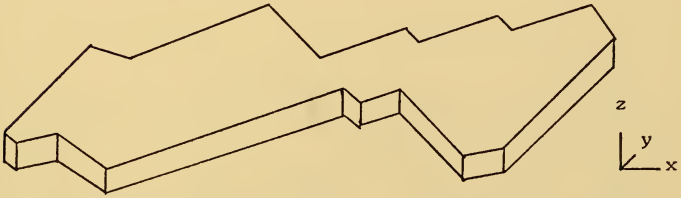


Figure 28b. Microstate 100, z = 2 plane (70 atoms).

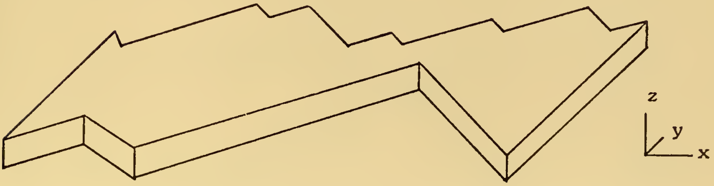


Figure 28c. Microstate 120, z = 2 plane (66 atoms).

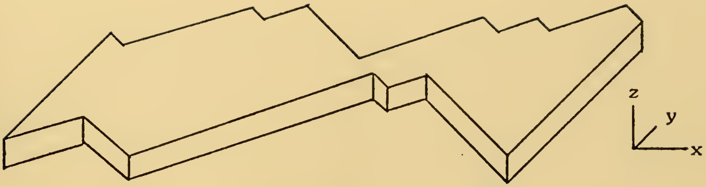


Figure 28d. Microstate 130, z = 2 plane (66 atoms).

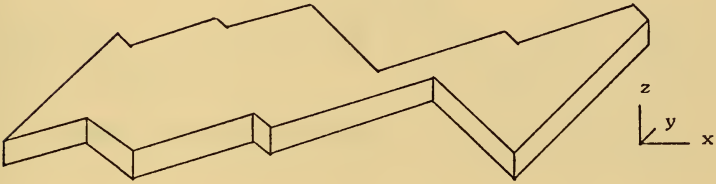


Figure 28e. Microstate 140, z = 2 plane (64 atoms).

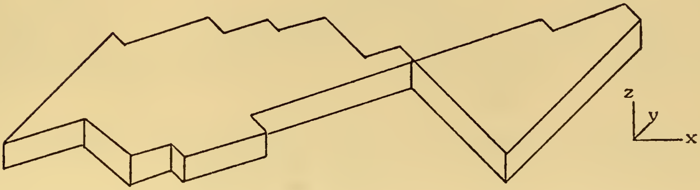


Figure 28f. Microstate 150, z = 2 plane (63 atoms). The two groups of atoms are held together only by one NN2 interaction.

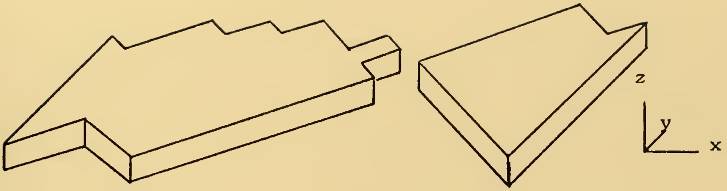


Figure 28g. Microstate 170, $z = 2$ plane (61 atoms).

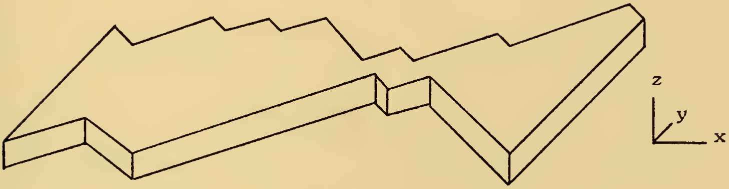


Figure 28h. Microstate 180, $z = 2$ plane (61 atoms).

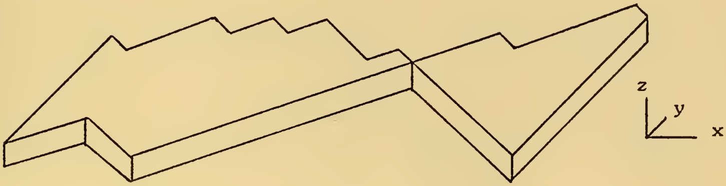


Figure 28i. Microstate 190, $z = 2$ plane (60 atoms). The two groups of atoms are joined only by one NN2 interaction.

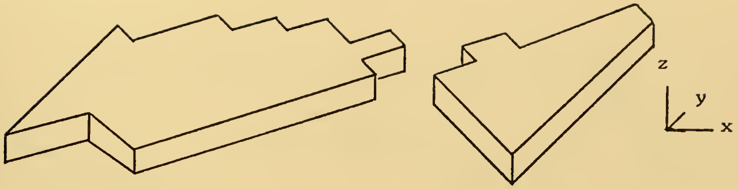


Figure 28j. Microstate 200, z = 2 plane (59 atoms).

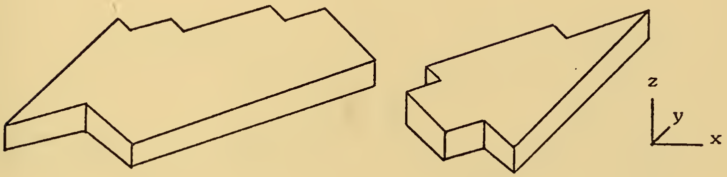


Figure 28k. Microstate 240, z = 2 plane (57 atoms).

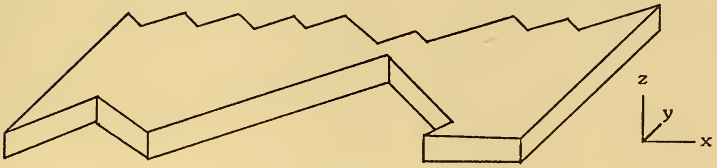


Figure 28l. Microstate 290, z = 2 plane (63 atoms - i.e., some atoms have fallen down from the z = 3 plane).

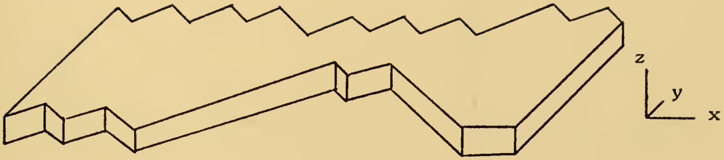


Figure 28m. Microstate 330, z = 2 plane (61 Atoms).

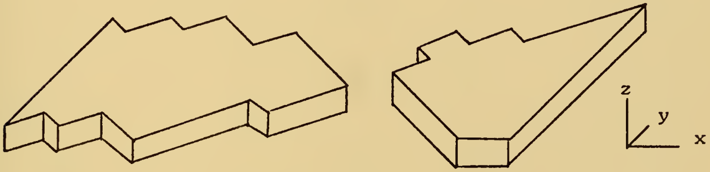


Figure 28n. Microstate 350, z = 2 plane (57 atoms).

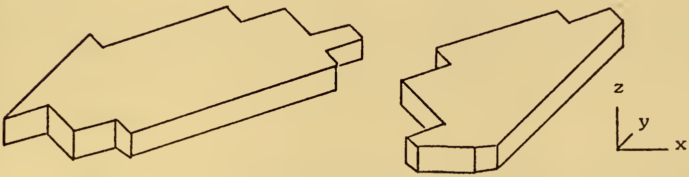
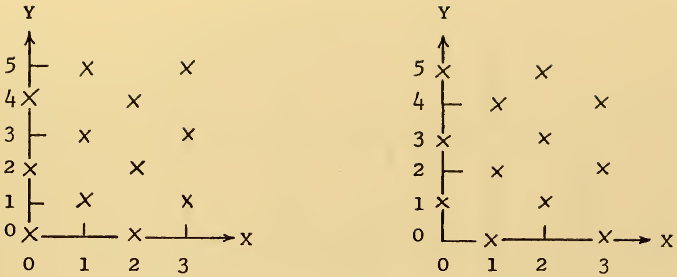


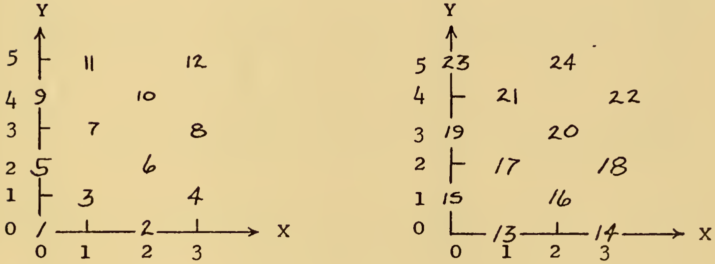
Figure 28o. Microstate 370, z = 2 plane (55 atoms).



(a) z even

(b) z odd

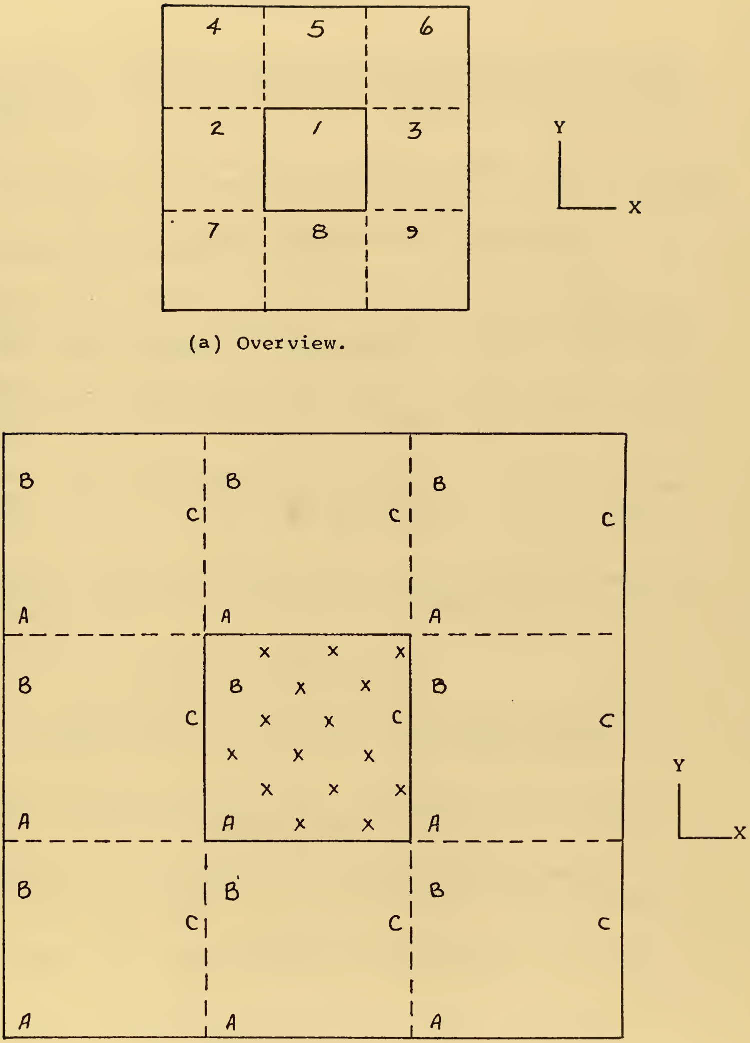
Figure 29. Relative Atomic Positions in a Lattice Plane. Here, IX = 4 and IY = 6.



(a) z = 0

(b) z = 1

Figure 30. The Site Numbering Scheme Imposed by the Lattice Generator. Here, IX = 4 and IY = 6.



(a) Overview.

(b) Specific site images for IX = IY = 6, z even.

Figure 31. Visualization of Periodic Boundary Conditions.

# BIBLIOGRAPHY

1. Anderman, A., Computer Investigation of Radiation Damage in Crystals, Atomics International, Canoga Park, California, 1966.

2. Bennema, P., "The Importance of Surface Diffusion For Crystal Growth From Solution ," J. Crystal Growth, v. 5, p. 29, 1969.

3. Blatt, J.M., Introduction to Fortran IV Programming, p. 56, Goodyear Publishing Co., 1968.

4. Burton, W.K., Cabrera, N., and Frank, F.C., "The Growth of Crystals and the Equilibrium Structure of Their Surfaces," Phil. Trans., Royal Soc. of London, v. 243A, p. 299, 1951.

5. Gibson, J.B., Goland, A.N., Milgram, M., and Vineyard, G.H., "Dynamics of Radiation Damage," Phys. Rev., v. 120, p. 1229, 1960.

6. Gilmer, G.H., and Bennema, P., "Simulation of Crystal Growth with Surface Diffusion," J. Appl. Phys., v. 43, p. 1347, 1972.

7. Girifalco, L.A., and Weizer, V.G., "Application of the Morse Potential Function to Cubic Metals," Phys. Rev., v. 114, p. 687, 1959.

8. Harrison, D.E., private communication.

9. Jackson, K.A., Uhlmann, D.R., and Hunt, J.D., "On the Nature of Crystal Growth from the Melt," J. Crystal Growth, v. 1, p. 1, 1967.

10. Leamy, H.J., and Jackson, K.A., "Roughness of the Crystal-Vapor Interface," J. Appl. Phys., v. 42, p. 2121, 1971.

11. Marsaglia, G., and Bray, T.A., "One-Line Random Number Generators and Their Use in Combinations," Comm. of ACM, v. 11, p. 758, 1968.

12. Melissinos, A.C., Experiments in Modern Physics, p. 476, Academic Press, 1966.

13. Morse, P.M., "Diatomic Molecules According to the Wave Mechanics. II. Vibrational Levels," Phys. Rev., v. 34, p. 57, 1929.

14. Naylor, T.H., Balintfy, J.L., Burdick, D.S., and Chu, K., Computer Simulation Techniques, p. 47, John Wiley and Sons, Inc., 1967.

15. Wood, W.W., and Parker, F.R., "Monte Carlo Equation of State of Molecules Interacting with the Lennard-Jones Potential. I. A Supercritical Isotherm at About Twice the Critical Temperature", J. Chem. Phys., v. 27, p. 720, 1957.

## INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Documentation Center 2
Cameron Station
Alexandria, Virginia 22314

2. Library, Code 0212 2
Naval Postgraduate School
Monterey, California 93940

3. Professor Don E. Harrison, Jr. 3
Department of Physics Code 61Hx
Naval Postgraduate School
Monterey, California 93940

4. CPT Henry W. Sterbenz, Jr., USA 1
620 80th Street
Brooklyn, New York 11209

DOCUMENT CONTROL DATA - R & D

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

| | |
|---|---|
| 1. ORIGINATING ACTIVITY *(Corporate author)*<br>Naval Postgraduate School<br>Monterey, California 93940 | 2a. REPORT SECURITY CLASSIFICATION<br>Unclassified<br>2b. GROUP |

3. REPORT TITLE

A Computer Simulation of Copper Crystal Surface Dynamics

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Master's Thesis; June 1972

5. AUTHOR(S) *(First name, middle initial, last name)*

Henry William Sterbenz, Jr.

| 6. REPORT DATE | 7a. TOTAL NO. OF PAGES | 7b. NO. OF REFS |
|---|---|---|
| June 1972 | 141 | 15 |

| | |
|---|---|
| 8a. CONTRACT OR GRANT NO.<br>b. PROJECT NO.<br>c.<br>d. | 9a. ORIGINATOR'S REPORT NUMBER(S)<br>9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* |

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

| | |
|---|---|
| 11. SUPPLEMENTARY NOTES | 12. SPONSORING MILITARY ACTIVITY<br>Naval Postgraduate School<br>Monterey, California 93940 |

13. ABSTRACT

A simple physical model of a copper crystal surface was developed. Atoms were considered as quasi-hard spheres which occupied perfect lattice positions. A computer simulation, based on energy consideration only, using the Monte Carlo method was developed, tested and used to study equilibrium surface microstates. As a result of this study, four conclusions were drawn:

1. This model holds promise for further investigation of real crystal surface phenomenon.
2. Minimum energy considerations cause atoms to align themselves preferentially in a ⟨110⟩ direction on the surface of a face-centered cubic crystal.
3. Stepped surface configurations are fairly stable, but isolated "stub" atoms and vacancies tend to coalesce with other "stubs" and vacancies, respectively.
4. Random motions of the individual atoms cause aggregates of atoms to break apart and recombine.

| KEY WORDS | LINK A | | LINK B | | LINK C | |
|---|---|---|---|---|---|---|
| | ROLE | WT | ROLE | WT | ROLE | WT |
| Copper Surface Simultation | | | | | | |
| Copper Surface Microstates | | | | | | |
| Surface Simulation | | | | | | |
| Surface Microstates | | | | | | |

Sterbenz

A computer simulation of copper crystal surface dynamics.

Thesis 134730
S705 Sterbenz
c.1 A computer simulation of copper crystal surface dynamics.