YAQUI User's Manual for Fireball Calculations

by

J. L. Norton
H. M. Ruppel

Los Alamos scientific laboratory
of the University of California
LOS ALAMOS, NEW MEXICO 87545

An Affirmative Action/Equal Opportunity Employer

UNITED STATES
ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
CONTRACT W-7405-ENG. 36
This work was supported by the Defense Nuclear Agency.
CONTENTS

I. DESCRIPTION OF CODE
A. Initial Zoning .................................................. 1
B. Fireball Initialization ........................................ 1
C. Particle Setup .................................................... 2
D. Particle Movement ............................................... 4
E. Turbulence ........................................................ 5
F. Rezone Techniques .............................................. 6
G. Mixed Equation of State ....................................... 12
H. System Improvements ......................................... 12

II. DESCRIPTION OF INPUT .............................................. 13

III. RESULTS OF A SAMPLE CALCULATION
A. Initial Conditions ............................................... 17
B. Turbulence Seeding Conditions ............................... 25
C. Turbulence Equilibrium Conditions ....................... 27
D. Moderately Late-Time Conditions
   (Eight Torus-Formation Times) ............................ 30

IV. FLOW DIAGRAM .................................................. 45

V. CODE LISTING ..................................................... 47
A. LTSS-Dependent Code ......................................... 47
B. CROS-Dependent Code ......................................... 61
C. KRONOS-Dependent Code ..................................... 66
D. KRONOS/CROS-Dependent Code ............................. 69
E. System-Independent Code .................................. 76

REFERENCES ......................................................... 297
YAQUI USER'S MANUAL FOR
FIREBALL CALCULATIONS

by

J. L. Norton and H. M. Ruppel

ABSTRACT

Recent modifications and additions made to the YAQUI code are described. This code, which was written to simulate nuclear explosions in the atmosphere, has been improved to include the effects of turbulence. In addition, it now allows input data to be obtained by direct interpolation in the one-dimensional results of early time radiation codes. This new version also makes most of the input free-format (namelist) and is composed of modules for easier modification and isolation of computer system dependence.

I. DESCRIPTION OF CODE

The YAQUI code is a combination of two fluid-dynamical techniques, ALE and ICE. ALE is an acronym for Arbitrary-Lagrangian-Eulerian; using this method the finite difference mesh points can be moved with the fluid (Lagrangian), held stationary (Eulerian), or moved by some prescribed rezone algorithm (Arbitrary). ICE stands for Implicit Continuous-fluid Eulerian; because the hydrodynamic equations are solved implicitly, the technique can be used to solve for flow at all speeds. The analyses of these techniques were presented by Harlow and Amden1 and by Hirt et al.,2 and an initial version of the code was described by Amden and Hirt.3

This section will examine the differences between the initial version of the code and the modified version used to do the calculations in this report.

A. Initial Zoning

An initial YAQUI zoning setup is shown in Fig. 1. The initial grid is determined by the parameters DR, DZ, IBAR, JBAR, and YB. First, a uniform grid of IBAR zones in the r-direction and JBAR zones in the z-direction with the bottom at z = YB is generated. The zones are all DR cm wide by DZ cm high. Then the parameters FREZXR, FREZYB, and FREZYT are examined. If any one of them is larger than 1, it is assumed that a region of nonuniform zoning will surround the uniform region. The parameters that describe the nonuniform region are IUNF, JUNF, JCEN, and REZY0. IUNF is the number of zones starting at the left that are to remain uniform. To the right of this region, the zone widths are related by

DR(I+1) = DR(I)*FREZXR ,

where I increases to the right. Likewise, JUNF is the number of zones in the z-direction that will remain uniform. Referring to Fig. 1, one can see that the region of uniform zoning will be centered on the line JCEN zones above the bottom of the grid with half the uniform zones above the top of the JCENth zone and half below. The heights of the zones above the region of uniform zoning will be related by the equation

DZ(J+1) = DZ(J)*FREZYT ,

where J increases toward the top. The heights of the zones below the region of uniform zoning will be related by the equation.

1
B. Fireball Initialization

The initial version of the code read in data that had been interpolated from one-dimensional spherical form to a two-dimensional grid with velocities centered at cell edges as in MAC. However, in YAQUI, velocities appear at vertices; therefore, a second interpolation was required. To avoid this unnecessary step and the resultant smoothing, the code was modified to permit inputting data in one-dimensional form. The data is then interpolated directly onto the YAQUI grid.

The interpolation technique is as follows. The 1D data is input one zone per card. Card i contains the radius of the outer boundary of zone i (r_i), the velocity of the outer boundary (v_i), and the specific internal energy and density of the zone (\epsilon_i and \rho_i, respectively) in the format 4E15.0.

The top of the JCE zone is redefined to be at the point z = REZY0. Because FREZYB will move the bottom of the mesh, it is necessary to recalculate YB based on JUNF, JCEN, REZYO, and FREZYB. In defining grids for problems with the ground at the bottom, care must be taken to calculate FREZYB so that the bottom of the J = 2 zone coincides with y = 0.

The grid is surrounded on three sides by layers of fictitious cells, shown as dotted lines in Fig. 1. These allow a variety of boundary conditions to be applied. The bottom layer of fictitious cells causes the first row of real zones to be at J = 2 rather than at J = 1. This fact can be confusing if one is not aware of the convention. The grid lines are known as I and J lines; I = 1 is at the extreme left; J = IBAR + 2 is at the extreme right of the dummy column; J = 1 is at the bottom of the bottom fictitious cells; and J = JBAR + 3 is at the top of the top fictitious cells. The zone whose lower left corner is at the intersection of the lines I and J is known as zone (I,J).

To carry out the interpolation, one divides each spherical shell into subzones using both radial and angular segments. Consider a portion of a 1D zone as shown in Fig. 4. Of course, in practice

![Fig. 1. Sample initial zoning setup.](image1)

![Fig. 2. Sample 1D fireball initial data configuration.](image2)
Fig. 3. Superposition of 1D fireball data on the 2D YAQUI grid.

Fig. 4. Division of 1D fireball data into subzones for interpolation purposes.

Fig. 5. A typical 1D Interpolation subzone.

$\theta_1$ is 0° and $\theta_2$ is 180°. $\Delta \theta$ is determined from $180^\circ/NTH$, where $NTH$ is an input quantity; $\Delta r$ is determined from $\frac{r_i - r_{i-1}}{NRAD}$, where $NRAD$ is also an input quantity. $NTH$ and $NRAD$ are held constant for all zones. The center of a subzone is defined as shown in Fig. 5. The interpolation procedure, then, is as follows. Consider a single subzone of a spherical shell. Find into which YAQUI zone the center of the subzone falls. Assign all the mass, momentum, and internal energy of the subzone to the YAQUI zone. One can estimate the accuracy of the procedure by accumulating the volumes of the subzones that are assigned to an individual YAQUI zone. When all of the subzones have been assigned, the volume of the YAQUI zone and the sum of the volumes of the subzones assigned to the YAQUI zone should be nearly the same except for those YAQUI zones only partially within the outermost spherical shell,
To handle the latter, the sum of the subzone volumes is also used as a flag. If the volume error \( \frac{(\text{sum of subzone volumes} - \text{YAQUI zone volume})}{\text{YAQUI zone volume}} \) is greater than 1%, then the five nearest cells on the top, bottom, and right are examined to see if any are within the fireball. (See Fig. 6.) If the YAQUI zone falls only partially within the fireball, at least one of the five neighbors will not be within the fireball at all. If all of the five neighbors are fireball cells, then there is an internal volume inaccuracy and the interpolation must be made finer (\( \Delta r \) and \( \Delta \phi \) must be decreased). If a nonfireball neighbor is found, then the specific internal energy and density of this neighbor are used to establish the mass and internal energy of the part of the YAQUI zone that is not within the fireball.

To understand this better, consider the example shown in Fig. 6. Let \( r_{\text{max}} \) be the outer radius of the last ID zone. The horizontally crosshatched portion of zone \((i,j)\) is within the fireball and the vertically crosshatched portion is outside of the fireball. The relative volume error will certainly be greater than 1%. Of the five neighbors shown, cell \((i+1,j-1)\) is completely outside of the fireball; its specific internal energy and density would be used to establish the internal energy and mass of the vertically crosshatched portion.

Once the internal energy, mass, and momentum components are established for all YAQUI zones that fall completely or partially within the fireball data, the specific internal energy is determined by dividing the zone internal energy by the zone mass. The zone density is determined by dividing the zone mass by the zone volume. The velocities are vertex quantities and are determined as follows. The cells containing a particular vertex are examined to see if all are fireball cells, i.e., if the vertex lies within the fireball. If any are not, then the vertex velocity is set to zero. If all are fireball cells, then the vertex takes \( \frac{1}{N} \) of the momentum of each neighbor (for central cells, \( N = 4 \); for boundary cells, \( N = 2 \)).

C. Particle Setup

The particle setup has two options: a rectangular particle region and a circular particle region.

Rectangular Region

The lower left-hand corner of the particle region is at \((X_C,Y_C)\) and the upper right-hand corner is at \((X_D,Y_D)\). The actual location of the particles is determined by superimposing a uniform grid of zones, which are \( \text{DRPAR} \) wide by \( \text{DZPAR} \) high starting at \((X_C,Y_C)\) and placing a particle at the center of each zone if the particle falls within the rectangular region. See Fig. 7.

Circular Region

A circular particle region is more useful for fireball problems. This option is initiated by setting \( Y_D = 0 \). The circular region's center is at \((0,Y_C)\), its radius is \( X_D \) (\( X_C \) is not used). Particles are placed as in the rectangular region except that the uniform grid starts at \( (0,Y_C-X_D) \) and only particles that fall within the circle are used.

![Fig. 6. Technique for handling YAQUI zones only partially within the fireball.](image)

![Fig. 7. Particle generation scheme.](image)
D. Particle Movement

Calculating with nonrectangular cells poses some additional problems in the movement of marker particles. The approach used in the initial version of YACUI was to define a grid of particle cells with constant $\Delta x$ and $\Delta y$ to overlay the calculational mesh. Masses and momenta were obtained by linear interpolation in the values assigned to the calculational grid for each vertex of the particle grid. The final step was to interpolate in these particle grid values to find the particle velocity with which it was to be moved. In addition to requiring two interpolations each cycle, this method tends to break down when a disparity exists between the size of calculation and particle cells. If the variable resolution causes large cells in one region of space and small cells in another, accuracy will be severely restricted. For such a situation there may be many fluid cells for one particle cell, or vice-versa. The former will cause smoothing of the velocity field as applied to the particles, the latter, to an uncertain determination.

To take advantage of the greater resolution that variable cells allow, we dispensed with the concept of an overlay grid, and interpolated directly in the fluid field. For this approach, two points must be considered: you must know in which cell a marker particle lies, and you must implement a reasonable scheme for interpolation in a skewed mesh. Consider first the method of searching for the values with which to do the interpolation. This is done by drawing vectors in succession from the particle to each of the vertices of a calculational cell. If these proceed in order and the angles between successive vectors are less than $\pi$ for each of the four vectors, the particle lies within the cell. This may be seen more clearly in Fig. 8. In Fig. 8(a) the particle lies inside the cell. In Fig. 8(b), taking the vectors in order, the vector to vertex 4 lies between the vectors to vertices 2 and 3, and hence the particle lies outside the cell. In Fig. 8(c) the angle between the vectors to vertices 4 and 3 is greater than $\pi$, and the particle lies outside the cell.

The numbering of vertices is arbitrary and may be clockwise or counter-clockwise, but must be consecutive. The rationale for this seemingly complicated criterion is that it avoids precise knowledge of any angles and hence requires no reference to trigonometric subroutines. Improved methods for which this is also true have since been found, one of which we will mention later. However, in the current version of the code, the above approach is used. To increase efficiency a one-dimensional array holds the number of the cell containing a particle at the beginning of a calculational cycle.

If a particle is no longer in the cell in which it began the cycle, an efficient search is undertaken to find the new location. For skewed cells, the particle can move more than one cell away, and hence determining an optimum trajectory along which to search is useful. This minimizes
the number of tests that must be made.

Two criteria are imposed in selecting the interpolation scheme once the appropriate cell has been identified:

1. The interpolated velocity must be bounded by the vertex values — that is, its value must lie between the smallest and largest of the four velocities.

2. The interpolated values must be continuous across cell boundaries.

An efficient method is to map the (x,y) space to the logical (ν,θ) space by the transformation

\[ \mathbf{x} = (1-\eta) (1-\theta) \mathbf{x}_1 + (1-\eta) \theta \mathbf{x}_2 + \eta \theta \mathbf{x}_3 + (1-\theta) \eta \mathbf{x}_4. \] (1)

The vectors \( \mathbf{x}_1, \ldots, \mathbf{x}_4 \) are drawn from any convenient origin to the vertices of the cell. The vector \( \mathbf{x} \) points to the position for which interpolated velocities are desired. From the knowledge of the coordinates \( \mathbf{x}_1, \ldots, \mathbf{x}_4, \) and \( \mathbf{x} \), we can invert Eq. (1) to obtain the values of \( \eta \) and \( \theta \) corresponding to the point \( (x,y) \). In the \( (\nu,\theta) \) space we then do a bilinear interpolation, using for any scalar the form

\[ s(\eta,\theta) = (1-\eta) (1-\theta) s_1 + (1-\eta) \theta s_2 + \eta \theta s_3 + (1-\theta) \eta s_4. \] (2)

We search for the new location of a particle by drawing a line from the center of the cell in which the particle began the cycle to its present location. This defines a trajectory that may pass through several cells and along which we perform our test for particle in cell. Although this is somewhat cumbersome, it requires testing very few cells, usually just one or two. An improved approach used by Pracht\(^4\) eliminates the separate testing and automatically defines the direction of search. If the particle lies in the cell in question, the values of \( \eta \) and \( \theta \) obtained from Eq. (1) will both lie between 0 and 1. If either does not, the cell indicated by the values of \( \eta \) and \( \theta \) is examined. For example, if \( \eta = 1.2 \), increase \( \eta \) by one and recalculate \( \eta \) and \( \theta \).

This approach is neater and more efficient. In future versions of the particle transport, we plan to incorporate such a scheme into YAQUI.

An additional component of particle displacement is required when the effects of turbulence are being calculated. This is described in the next section.

E. Turbulence

A significant new feature in the present version of YAQUI is the addition of a one equation transport model for turbulence. The modified equations, which we now solve are the mass equation

\[ \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \] (3)

the momentum equation

\[ \frac{\partial \rho \mathbf{u}}{\partial t} + \frac{\partial}{\partial x_j} (\rho \mathbf{u} \mathbf{u}_j) + \rho \mathbf{v} \cdot \nabla \mathbf{u} = \nabla \cdot \mathbf{P} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) \] (4)

the internal energy equation

\[ \frac{\partial e}{\partial t} = \rho \mathbf{v} \cdot \mathbf{u} + \frac{2}{3} \rho \mathbf{v} \cdot \mathbf{v} + \rho \mathbf{v} \cdot \nabla e \] (5)

and the turbulence energy equation

\[ \frac{\partial \epsilon}{\partial t} = \frac{\partial}{\partial x_j} \left( \nu \frac{\partial \epsilon}{\partial x_j} \right) - \frac{2}{3} \frac{\partial}{\partial x_j} \left( \sigma \mathbf{v} \cdot \mathbf{v} \right) \] (6)

The turbulence viscosity is given by

\[ \sigma = \beta \nu \sqrt{\epsilon}. \] (7)

For conciseness we have used the following definitions:

\[ \Pi_{ij} = (\mu + \rho \sigma) \epsilon_{ij} + \partial \epsilon_{ij} + \delta_{ij} \left[ (\frac{2}{3} \rho \sigma) \frac{\partial u_k}{\partial x_k} \right] \] (8)

\[ H_{ij} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}, \] (9)

\[ \Gamma = \epsilon_{ij} \frac{\partial u_i}{\partial x_j}, \] and

\[ e_{ij} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}. \] (11)
The parameters of the model are \( \beta \) and the scale, \( s \). Values are obtained phenomenologically from fits to experiment. For \( \beta \) we use the constant, \( \beta = 0.02 \). For the scale, constant values over the mesh and the more complicated phenomenological form

\[
s = 0.14 \, d(z) \left( 2 - \frac{v(r,z)}{v(0,z)} \right) \quad v(r,z) > 0 \quad (12a)
\]

\[
= 0.28 \, d(z) \quad v(r,z) < 0 \quad (12b)
\]

have been used. The function \( d(z) \) is the radial distance to the point at which the velocity changes sign, and \( v(0,z) \) is the axial component of the velocity on the axis, \( r = 0 \). In a skewed mesh Eq. (12) requires considerable logic and interpolation. Because \( s \) is only crudely known, we generally approximate the above form by

\[
s = 0.14d \left( 2 - \frac{v(r,z)}{v(0,z)}_{\text{max}} \right) ; \quad (13)
\]

in this simplified relation, \( d \) is a constant, usually taken to be 1, and \( v(0,z)_{\text{max}} \) is the maximum value of \( v \) along the axis. This expression may overestimate the scale for fireballs and cut down on the rate of decay of the turbulence energy. Erring in this direction gives us an upper bound on the effect of the turbulence. In the latest version of the code, we simply incorporate a constant scale throughout the mesh, though provision exists in the code for a more general treatment.

In differencing the turbulence additions, we can take advantage of much of the existing framework of the basic YAQUI program. Because small time-level inconsistencies in the turbulence equations are insignificant, economy is a major criterion in establishing the order in which the several additions are included. For example, looking at the structure of the modified stress tensor, we are led to replace \( \lambda + \lambda + \rho \sigma, \lambda + \lambda - \frac{2}{3} \rho \sigma \) and \( \rho + \rho + \frac{2}{3} \rho q \). Similarly \( H_{ij} \) is calculated from \( \rho \) and \( \rho \) at the old time level; this allows the stress tensor to be handled explicitly in Phase 1 simplifying the addition. The term \( \frac{3}{at} \rho \sigma \tau_i \) in the momentum equation is modeled, using the flux approximation, as \( -\frac{3}{at} \rho \frac{\partial}{\partial x_i} \); this requires saving the two components of \( \sigma \frac{\partial}{\partial x_i} \) for each cell from the previous time step. Clearly such an approach centers the time derivative about \( t - \frac{\delta t}{2} \) and not at the same time level as \( H_{ij} \). For reasonable time variations, the inconsistency should not be important.

Three more diffusion terms must be differenced: one in the mass equation, one in the internal energy equation, and one in the turbulence energy equation. Each of these is added explicitly in Phase 1 with derivatives from time level \( n \).

The transport equation for the turbulence energy \( q \) is mainly handled explicitly. However, we can include some advanced time information by writing Eq. (6) in the form

\[
\left( \rho q \right)^{n+1} \left[ 1 + \frac{2}{3} \frac{\partial}{\partial t} + \frac{2 \sigma}{\beta a^2} \right] = \left( \rho q \right)^n - \sigma t \left[ \frac{\partial \rho}{\partial \rho} \right. \\
\left. + \frac{2}{3} \rho \left( \rho \frac{\partial \rho}{\partial \rho} \right)^2 \right] + \left( \rho \frac{\partial \rho}{\partial \rho} \right)^n . \quad (14)
\]

In cylindrical geometry \( \Gamma \) takes the form

\[
\Gamma = 2 \left[ \left( \frac{a \partial u}{\partial t} \right)^2 + \left( \frac{a \partial v}{\partial z} \right)^2 + \frac{1}{2} \left( \frac{a \partial u}{\partial z} + \frac{a \partial v}{\partial t} \right)^2 + \left( \frac{a \partial u}{\partial r} \right)^2 \right] . \quad (15)
\]

All the terms on the right-hand side of Eq. (14) are evaluated at time level \( n \) as cell-centered quantities. For most of the terms this proceeds naturally; but for the buoyancy creation, \( \rho \frac{\partial \rho}{\partial \rho} \), a small reach is necessary. In the notation of Fig. 9 we could evaluate the derivatives directly at the

![Fig. 9. Quantities involved in YAQUI derivative evaluation.](image)
vertices 1–4 and then average to obtain \((V_p \cdot V_p)_o\). This uses information at nine cells and is unnecessarily smoothing. For this reason we use cells a, b, c, and d and obtain the derivatives at the center of cell o in terms of these. For example,

\[
\frac{\partial \phi}{\partial r} = \frac{1}{2A} \left\{ (\rho_a - \rho_c) (z_b - z_d) + (\rho_b - \rho_d) (z_c - z_a) \right\},
\]

where \(A\) is the area of the quadrilateral abcd. We are particularly concerned about smoothing out derivatives because the sensitivity of the turbulence energy and the level of turbulence that can be supported depend on the size of gradients of various quantities. Derivatives of vertex quantities are calculated at cell centers by the equations

\[
\begin{align*}
\frac{\partial u}{\partial r} &= \frac{1}{2A} \left\{ (z_2 - z_4) (u_1 - u_3) + (z_3 - z_1) (u_2 - u_4) \right\}, \\
\frac{\partial u}{\partial z} &= \frac{1}{2A} \left\{ (r_2 - r_4) (u_1 - u_3) + (r_1 - r_3) (u_2 - u_4) \right\},
\end{align*}
\]

where \(A\) is the area of the cell.

If we expand the derivatives about the center of the cell, we find that the error is proportional to second and higher derivatives. In a similar fashion we can obtain derivatives of cell-centered quantities at cell centers (as discussed above) by appropriate choice of configurations. The necessity to optimize the calculation of gradients implies that high-resolution calculations are important in evaluating turbulence models.

Similar problems arise for the Laplacian or diffusion-like terms. Again, several approaches are possible and in this case optimization is much less crucial, because the level of turbulence is only indirectly affected. The criteria applied to select an approach for differencing such terms were: that an equation of the form

\[
\frac{\partial c}{\partial t} + \nabla \cdot \nabla c = \nabla \cdot \sigma \nabla c
\]

could not lead to negative \(c\) anywhere in the mesh, and that the difference form of \(\nabla \cdot \nabla c\) should reduce to the expected form for the case of a uniform rectangular mesh. For example, if \(c\) represents the concentration of a chemical species,

\[
\frac{1}{V_o} \sum_{j=1}^{4} \frac{c_j - c_o}{2} d_{jo}^2 <r_{jo} > <o_{jo} > = (V_o \cdot V_c)_o,
\]

where \(c_j\) is the concentration in cell \(j\), \(d_{jo}\) is the length of the cell side between cells \(o\) and \(j\), \(<r_{jo}\>\) is the distance between the centers of cells \(o\) and \(j\), \(A_j\) is the area of cell \(j\), \(V_o\) is the volume of cell \(o\), and \(<o_{jo}\>\) is \(\frac{1}{2} A_o\). See Fig. 10 for examples. In Cartesian coordinates for a uniform rectangular mesh with sides \(\delta x\), \(\delta y\), and constant \(\sigma\), this reduces to

\[
c \left( \frac{c_{1+3} c_{3-2} c_{2+4} - c_{2+4} c_{3-2} c_{1+3}}{\delta x^2 + \delta y^2} \right)
\]

as we would wish. The vertices of this quadrilateral are at the center of their respective cells.

In the finite difference approach with finite time steps it is possible for the turbulence energy to become negative in certain cells. If, for example, \(\frac{\partial}{\partial t} \nabla \cdot \nabla p\) is positive and greater than \(\partial q\) in a given cell, more energy will be subtracted from the cell than it contains. One could reduce the time step, but this would never really solve the problem. The assumption we make is that numerically, turbulence can go negative in regions in which it is decaying, if \(q < 0\), we set \(q\) to zero. Experience has shown for the fireball calculations that less than 1% of the turbulence energy is lost by this procedure.

![Fig. 10. Definition of quantities used in differencing of turbulence part of the concentration equation.](image-url)
As far as convection is concerned, the quantity $pq$ is convected and, because the difference equations for the convection conserve the convected quantity identically, the turbulence energy is conserved in Phase III. The total energy, however, is not. In the initial code version\textsuperscript{3} an equation for the total energy is written, and this is the quantity that is convected. Momentum is also convected, and hence conserved; the internal energy is obtained by subtracting the kinetic energy from the total energy. This places all the uncertainty in the internal energy, which if it is a small part of the total energy, can lead to large fractional errors in the internal energy.

With the introduction of the equation for turbulence energy, it has seemed convenient to deal directly with the internal energy. In the code the equation for the internal energy follows the pressure iteration, allowing us to calculate the $pdV$ work with a time-advanced pressure. The quantities that are convected in Phase III, then, are the internal energy, the turbulence energy, and the momentum, which are individually conserved. In general, this means that the kinetic energy and, thus, the total energy, $E = \rho T + \rho u^2 + pq$, will not be conserved. This lack of energy conservation does not seem to be significant and is ignored in actual calculations.

A small change has been made in the energy equation related to the smoothing of the velocity field for computational stability. If alternate mesh vertices are not coupled in some way (see Ref. 3), an instability arises. The approach we elect is to couple the alternate nodes only when a local minimum or maximum in the velocity field occurs. We apply a restoring acceleration to vertex 4 of the form

$$\frac{1}{a_{nc}} \delta t \left[ \frac{1}{4} \left( u_4^r + u_3^r + u_6^r + u_8^r \right) - u_4^r \right]$$

(20)

(See Fig. 2 of Ref. 3).

In Ref. 3 this is applied to each vertex each cycle to control the instability. In our version we apply the restoring force only to those components of $\mathbf{u}$ at vertex 4 that have values larger than or smaller than any of the neighbors (1,3,6, and 8). That is, a local maximum or minimum in the $r$-component of $\mathbf{u}$ is smoothed by a restoring acceleration in the $r$-direction and similarly for the $z$-component. This is less diffusive than the approach that applies it everywhere each cycle. The intent is that when no instability threatens, this will not smooth gradients.

This node coupler clearly reduces the kinetic energy of the system, acting like a viscous dissipation. If this loss is ignored, it will lend to a gradual diminishing of the system's energy. On the other hand, to include the energy in the internal energy equation as viscous heating really has no basis in physical reality. However, in our version, we choose the latter option and include the energy removed by the node coupler in the equation for the internal energy. In the original version of the code the other choice was made.

One can see by examining the model equations for turbulence that no mechanism has been built in to initiate the turbulence spontaneously. That is if there is no turbulence present, i.e. $q=0$, none can be created or can grow. For this reason, the turbulence must be seeded initially and allowed to equilibrate with the mean flow through the creation and decay terms in the equation for the turbulence energy. Several alternative seedings have been tried and found to lead to the same turbulent configurations after a fairly short time. A very reasonable approach is to seed the turbulence proportional to the vorticity of the mean flow field. This is done after the field has been established, that is, shortly before torus formation time. This timing is not crucial. It can be seeded earlier and find its way to a similar level and distribution in a short time. We miss any high-intensity early time turbulence, likely initiated by Taylor instabilities as the device case and the very hot debris decelerate. We assume that the fluctuations decay in a few seconds because nothing appears to be present to support them. The equilibrium turbulence we calculate really only has meaning at later times.

Because the measured properties of the fireball, rate of rise and radial expansion, depend on the positions of the marker particles, it was felt that their motion should be coupled directly into the turbulence. This is done by adding a random turbulent diffusion velocity to the particle motion by
the following technique.

The diffusion of mass as a function of time from a point source of unit mass at position $\mathbf{r}_0$ is described by the diffusion equation

$$\frac{\partial \rho}{\partial t} = \lambda \nabla^2 \rho + \delta(\mathbf{r} - \mathbf{r}_0),$$

(21)

where $\rho$ is the density and $\lambda$ is a constant diffusion coefficient. Define the quantities $\delta r = \mathbf{r} - \mathbf{r}_0$, $\delta x = x - x_0$, $\delta y = y - y_0$, and $\delta z = z - z_0$. If $\rho(\delta r, t)$ is written as $\rho(\delta x, t) = X(\delta x, t) Y(\delta y, t) Z(\delta z, t)$, it can be shown that

$$X(\delta x, t) = \frac{1}{\sqrt{4\pi \lambda t}} e^{-\delta x^2/4\lambda t},$$

(22)

with similar expressions for $Y$ and $Z$.

In calculating the additional particle motion due to turbulence, consider the particle at time $t_o = 0$ to be a massless point at $\mathbf{r}_0$ and use Eq. (22) as a probability distribution function for determining the position of the particle at time $t = t_o + \delta t$. The turbulence viscosity then serves as the diffusion coefficient $\lambda$. In cylindrical coordinates one should solve Eq. (21) in a cylindrical basis

$$\rho(\delta r) = R(\delta r) Z(\delta z) .$$

However, $R$ cannot be determined in closed form and for small $\delta t$ and $\delta x$, with $\delta x > \frac{\lambda \delta t}{x_0}$, $R$ reduces to the form of $X$ in Eq. (22). Thus, Eq. (22) can be used with confidence in cylindrical geometry as long as it is applied in its region of validity. Furthermore, $\delta x$ must be small compared with a cell dimension so that the turbulence viscosity being used as the diffusion coefficient remains constant in the region and during the time considered.

The general Gaussian (normal) distribution is

$$f(y) = \frac{1}{\sqrt{2\pi} \sigma} e^{-y^2/2\sigma^2},$$

(23)

where $\sigma$ is the standard deviation. Equation (22) can be put in this form by setting

$$\sigma = \sqrt{2\lambda \delta t}$$

and

$$y = \delta x ,$$

(24a)

where we have replaced $t$ by $\delta t$.

The standard probability distribution function (random number generator) available on most computers is the uniform distribution $p(x)$,

$$p(x) = 1, \ 0 < x < 1$$

$$= 0, \ \text{elsewhere}. $$

(25)

What we need is the distribution given by Eq. (23) with $\sigma$ given by Eq. (24). To accomplish this Eq. (25) is mapped on to Eq. (23) (see Fig. 11) with the expression

$$p(x) \ dx = 2 f(y) \ dy,$$

(26)

because the probability of finding $x$ in $dx$ is equal to the sum of the probabilities of finding both $y$ and $-y$ in their respective $dy$'s. Therefore,
\[
\int_0^x p(x') \, dx' = 2 \int_0^y f(y') \, dy',
\]

or
\[
\int_0^x dx' = \frac{2}{\sqrt{\pi} \sigma} \int_0^y e^{-y'^2/2\sigma^2} \, dy'.
\]

With the variable transformation \( z' = \frac{x'}{\sigma} \),
\[
x = \frac{2}{\sqrt{\pi}} \int_0^y e^{-z'^2} \, dz'.
\]

By definition, the standard error function is
\[
erf(t) = \frac{2}{\sqrt{\pi}} \int_0^t e^{-t'^2} \, dt';
\]

therefore,
\[
x = erf\left(\frac{y}{\sigma}\right), \quad (27)
\]

and from Eq. (24),
\[
y = \sqrt{4\lambda \Delta t} \, erf^{-1}(x). \quad (28)
\]

The above is the mapping from Eq. (25) to Eq. (23) that is desired.

In principle, then, one determines a random number using Eq. (25) and maps it onto Eq. (23) using Eq. (28). One can see that the \( y \) in Eq. (28) is the required particle displacement due to turbulent diffusion if he remembers that the diffusion coefficient \( \lambda \) in Eq. (21) is replaced by the local turbulence viscosity.

One problem remains, namely, making the procedure efficient enough computationally so that one can afford to use it. The major difficulty is the calculation of inverse error functions. To do this without undue cost the following interpolation procedure was established.

The error function is monotonic increasing on the interval \([0,1]\); \( \text{erf}(0) = 0 \); and \( |\text{erf}(y)| \) approaches 1 asymptotically (see Fig. 12). Care must be taken in using \( \text{erf}^{-1}(x) \) for \( x \approx 1 \), for in this region \( \text{erf}^{-1}(x) \) is very large which could lead to an abnormally large particle displacement. To handle this, one selects a number \( W_{\text{MAXEF}} \) and distributes \( N_{\text{ERFV}} \) points equally spaced on the interval \([0, W_{\text{MAXEF}}]\) with point 1 located at 0 and point \( N_{\text{ERFV}} \) at \( W_{\text{MAXEF}} \). Thus, the equal spacing \( \Delta x \) will be
\[
\Delta x = \frac{W_{\text{MAXEF}}}{N_{\text{ERFV}}}, \quad (29)
\]

and the position of point \( i \), denoted by \( x_i \), will be
\[
x_i = (i-1)\Delta x. \quad (30)
\]

Now, let \( R(i) = \text{erf}(x_i) \). Then,
\[
x_i = \text{erf}^{-1}(R(i)) \quad (31)
\]

(see Fig. 12). Our problem is to find \( \text{erf}^{-1}(t) \equiv z \) given an arbitrary \( t \). To do this, we see that \( t = \text{erf}(z) \). Find an \( i \) such that
\[
R(i-1) < t < R(i) \quad \text{or} \quad \text{erf}(x_{i-1}) < t < \text{erf}(x_i). \quad (32)
\]

Therefore, \( x_{i-1} < \text{erf}^{-1}(t) < x_i \). One linearly interpolates to get
\[
\text{erf}^{-1}(t) \cong x_{i-1} + \frac{(t-R(i-1))}{(R(i) - R(i-1))} (x_i - x_{i-1}) \quad (32)
\]

or
\[
\text{erf}^{-1}(t) \cong \left[ (i-2) + \frac{(t-R(i-1))}{(R(i) - R(i-1))} \right] \Delta x \quad (32)
\]

using Eq. (30). One selects another random
number \( t \) if \( t > R(NERFV) \). Because \( R(1) = 0 \) and \( R(NERFV) = WMAXEF \) using Eq. (28), \( y \) is limited to the range

\[
0 < y < WMAXEF \sqrt{4A\Delta t} = WMAXEF \sqrt{2} \sigma.
\]

Thus, \( y \) is limited to \( WMAXEF \sqrt{2} \) standard deviations.

Because Eq. (22) is invalid near the axis of symmetry, another cutoff parameter \( RMINEF \) has been added. If the \( r \)-coordinate of the particle is \( \leq RMINEF \), the turbulent diffusion effect is not applied. In most cases, \( RMINEF \) can be considerably less than the \( \Delta r \)'s of the zones on the axis.

The current version of the code can be run easily with turbulence by-passed because the turbulence must be seeded initially. Most of the coding related to turbulence is excluded and no efficiency is lost if this version is used for turbulence-free calculations.

For a discussion of the output relevant to turbulence, see the sample calculation in Sec. III.

F. REZONE TECHNIQUES

The convection phase is appended to a Lagrangian calculation in a way that provides maximum flexibility for the continuous rezone. This is achieved by including the convective fluxes as functions of difference velocities,

\[
\overrightarrow{u_d} = \overrightarrow{u_{\text{fluid}}} - \overrightarrow{u_g},
\]

where \( \overrightarrow{u_g} \) is the grid velocity with which the mesh is moved in a given calculation cycle. For an Eulerian calculation \( \overrightarrow{u_g} = 0 \); for a Lagrangian calculation \( \overrightarrow{u_g} = \overrightarrow{u_{\text{fluid}}} \), the difference velocity vanishes, and there is no fluxing. In general the prescription to determine \( \overrightarrow{u_g} \) is at the discretion of the user. A general form that we have found useful and incorporated in the present version of the code is to write

\[
\overrightarrow{u_g} = \overrightarrow{u_{\text{fluid}}} + \frac{f}{\partial t} (\overrightarrow{\langle x \rangle} - \overrightarrow{x})
\]

for each vertex. That is, the grid velocity is composed of two components: the fluid velocity and a term to relax the mesh such that each vertex is at the average position of its nearest neighbors. The latter term prevents the mesh from distorting excessively. A typical value for \( f \) might be 0.05, which would relax the mesh in approximately 20 calculational cycles if there were no fluid motion. Many variations of the relaxation component are possible. The important point is to run as near to Lagrangian as possible to minimize the smoothing, but still to maintain some degree of regularity in the mesh.

G. MIXED EQUATION OF STATE

To model more accurately the atmospheric detonation of a Mylar balloon filled with methane, it was necessary to incorporate two equations of state in one problem, one for the combustion products and one for ambient air. The method was to divide the cells into two groups at \( t = 0 \), those inside the fireball and those outside. This division was done on the basis of the specific internal energy (\( e \)); those cells with \( e > 10^{10} \) erg/g were considered inside the fireball and all others, outside.

Define the concentration \( c_i \) as the ratio of the mass of constituent \( i \) in a given cell to the total mass of the cell. Obviously, if there are \( n \) constituents, \( \sum c_i = 1 \) in a given cell. For a problem like ours with only two constituents (exploded methane and ambient air), \( c_{\text{meth}} + c_{\text{air}} = 1 \) so that one only needs to keep track of \( c_{\text{meth}} \). Here we use \( c_{\text{meth}} \) to refer to the concentration of combustion products and \( \gamma_{\text{meth}} \) to refer to the effective \( \gamma \) for the combustion products.

Initially, \( c_{\text{meth}} \) is defined to be 1 in all fireball zones and 0 elsewhere. As the problem proceeds, the concentration is convected into the ambient region using

\[
\frac{\partial c_i}{\partial t} + \overrightarrow{u} \cdot \nabla c_i = 0.
\]

Note that this is just the Lagrangian form of the continuity equation.

To determine the effective \( \gamma \) in a given zone, we write

\[
\gamma_{\text{eff}} = c_{\text{meth}} \gamma_{\text{meth}} + (1-c_{\text{meth}}) \gamma_{\text{air}}.
\]

The pressure is then obtained from
\[ p = (y_{eff}^{-1})pe \]

**II. SYSTEM IMPROVEMENTS**

Because YAQUI is a rather large computer code, because many modifications have been made to it in the course of investigating various techniques, and because the Los Alamos Scientific Laboratory began the switch from batch to time-shared computing in the course of this research, considerable effort went into making YAQUI a convenient and flexible research tool. The various efforts involved are summarized below.

1. The original code was written in large blocks, which made it difficult to modify and also severely taxed the compilers and loaders. Thus, the code was divided into numerous smaller subroutines to avoid these problems. The concept of modularity was followed as closely as possible.

2. Because of rapidly shifting operating systems at LASL, it was important to make the code as system-independent as possible so that it could be easily switched from old to new systems. The code will now run on any of the three systems available at LASL, in either batch or time-sharing mode. It could be brought up on any other system with minimal difficulty, assuming sufficient small- and large-core storage were available.

3. Fixed format input is prone to error, whereas NAMELIST input tends to be system dependent. Thus, a NAMELIST input package was written that is system-independent for the most part, and can be easily modified for other systems.

4. Error checking within the code is meticulous. The code never assumes anything on the part of the user but monitors for errors, particularly in the setup phase, as though the user were completely unfamiliar with the code.

5. Because YAQUI runs often take several hours of CDC-7600 time, a problem may have to be run in several smaller pieces. For this reason, a flexible restart procedure was developed to allow the user to restart a problem at any point and change input parameters without having to actually modify the code itself.

6. The original YAQUI was very well documented externally in LASL report LA-5100. However, a code that is under heavy development is much easier to modify if it is carefully annotated internally. Work has been proceeding on this and is largely completed.

**II. DESCRIPTION OF INPUT**

The input to the code, except for the problem title and the 1D fireball initialization data, is in NAMELIST form. The basic rules are:

- Each input record begins with a $ in card column (cc) 2 followed immediately by the namelist name.
- Input values are of the form
  \[ \text{name} = \text{number}, \]
  where blanks may not occur within \text{name} or an individual number but are ignored elsewhere. \text{number} may be a single constant or a series of comma-delimited constants. Multipliers of the form \text{n*number} are allowed but not grouping with parentheses (e.g., \text{n*(n1, n2, n3)} is illegal).
- Continuation cards are legal. NAMELIST variable names and constants may not be split across card boundaries but hollerith fields may if they end in cc80 of one card and begin again in cc1 of the next card.
- An input record is terminated by a $ anywhere on the card except for cc1 and 2.
- Variables are stored without regard to type.
  If one has I = 5., I will contain a floating point 5. rather than an integer 5; likewise X = 2 will cause an integer 2 to be stored in X which will most likely be interpreted as a floating point 2.
- If a "P" is punched in cc1 of the first card of a namelist record, the entire record will be printed as part of the code output.

For most information about the namelist conventions, see the internal documentation at the beginning of the routine NAMLST in the listing of the code in Sec. V.

From here on, each input record will be described, the namelist name given, and the variable names listed and discussed.
**Record 1:** Namelist name - START

**Namelist Variables**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE (units)</th>
<th>Possible values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESTRT</td>
<td>LOGICAL</td>
<td>.TRUE., FALSE.</td>
<td>.TRUE.</td>
</tr>
<tr>
<td>FILM</td>
<td>LOGICAL</td>
<td>.TRUE., FALSE.</td>
<td>.TRUE.</td>
</tr>
<tr>
<td>PAPER</td>
<td>LOGICAL</td>
<td>.TRUE., FALSE.</td>
<td>.FALSE.</td>
</tr>
<tr>
<td>WRAPUP</td>
<td>REAL (s)</td>
<td>0.0&lt;WRAPUP&amp;2</td>
<td>20.0</td>
</tr>
</tbody>
</table>

(a) RESTRT - .TRUE. if the run will be a pickup from a previous dump tape. Otherwise it is an initial setup.

(b) FILM - .TRUE. if all output will go to film.

(c) PAPER - .TRUE. if all output with the exception of plots will go to paper.

(d) WRAPUP - time to allow for the last cycle, dumping plots, printouts, and general termination procedure. If the time limit is TLIM, then the calculation will be stopped and termination begun as soon as the run time exceeds TLIM-WRAPUP.

Example: $CARDN RESTRT = .TRUE., FILM = .TRUE., PAPER = .FALSE., WRAPUP '30.$

The next input depends on whether RESTRT is .TRUE. or .FALSE. Input based on RESTRT = .FALSE. (problem generation) will be considered first.

---

**Record 2:** Problem title, cc2-80 on one data card.

**Record 3:** Namelist name - CARDN

**Namelist Variables**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE (units)</th>
<th>Possible values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AO</td>
<td>Real (none)</td>
<td>0&lt;AO&amp;1.</td>
<td>0.1</td>
</tr>
<tr>
<td>AOFA5</td>
<td>Real (none)</td>
<td>0&lt;AOFA5&amp;0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>AOM</td>
<td>Real (none)</td>
<td>0&lt;AO&amp;1.</td>
<td>1.0</td>
</tr>
<tr>
<td>ANC</td>
<td>Real (none)</td>
<td>0&lt;ANC&amp;0.2</td>
<td>0.05</td>
</tr>
<tr>
<td>BO</td>
<td>Real (none)</td>
<td>0, 2.</td>
<td>0.0</td>
</tr>
<tr>
<td>CYL</td>
<td>Real (none)</td>
<td>0, 1.</td>
<td>1.0</td>
</tr>
<tr>
<td>DR</td>
<td>Real (cm)</td>
<td>0&lt;DR&amp;2</td>
<td>Must be specified</td>
</tr>
<tr>
<td>DT</td>
<td>Real (s)</td>
<td>0&lt;DT&amp;10^-3</td>
<td>10^-3</td>
</tr>
<tr>
<td>DTØ</td>
<td>Real (s)</td>
<td>0&lt;DTØ&amp;10^-3</td>
<td>1.0</td>
</tr>
<tr>
<td>DTØC</td>
<td>Real (s)</td>
<td>0&lt;DTØC&lt;10^30</td>
<td>1.0</td>
</tr>
<tr>
<td>DZ</td>
<td>Real (cm)</td>
<td>0&lt;DZ&lt;10^-5</td>
<td>Must be specified</td>
</tr>
<tr>
<td>EPS</td>
<td>Real (none)</td>
<td>0&lt;EPS&lt;1</td>
<td>10^-5</td>
</tr>
<tr>
<td>FREZXR</td>
<td>Real (none)</td>
<td>1&lt;FREZXR&lt;1</td>
<td>1.0</td>
</tr>
<tr>
<td>FREZYB</td>
<td>Real (none)</td>
<td>1&lt;FREZYB&lt;1</td>
<td>1.0</td>
</tr>
<tr>
<td>FREZYT</td>
<td>Real (none)</td>
<td>1&lt;FREZYT&lt;1</td>
<td>1.0</td>
</tr>
<tr>
<td>GR</td>
<td>Real (cm/s^2)</td>
<td>-10&lt;GR&lt;10^-5</td>
<td>0.0</td>
</tr>
<tr>
<td>GROVEL</td>
<td>Real (none)</td>
<td>0, 1, 2.</td>
<td>2.0</td>
</tr>
<tr>
<td>GZ</td>
<td>Real (cm/s^2)</td>
<td>-&lt;GZ&lt;980.0</td>
<td>980.0</td>
</tr>
</tbody>
</table>

AO - Amount of donor cell in momentum fluxing (0. is centered differencing; 1. is full donor cell).

AOFA5 - Stability condition uAt/Ax<AOFA5.

AOM - Amount of donor cell in mass fluxing (see AO).

ANC - Amount of node coupler (see Ref. 2).

BO - Interpolated donor cell coefficients; allows for partial cancellation of truncation errors in the convection terms.

CYL - Geometry-type switch. If CYL = 1., the calculation is done in cylindrical coordinates.
geom; if CYL = 0., it is done in slab geometry.

**DR** - Initial value of the width of the zones in the region of uniform zoning (see Sec. I.A).

**DT** - Initial time step.

**DTO, DTOC** - Two arrays that determine when plots and long prints (edits) will occur as a function of problem time. Edits will generally occur every DTO(I) seconds in the interval DTOC(I-1) < T < DTOC(I). DTOC(0) = 0.

**DZ** - Same as DR except zone height (see Sec. I.A).

**EPS** - Convergence criterion in the pressure iteration (see Ref. 2).

**FREZXR** - Geometric ratio of zone Ar's in the right region of nonuniform zoning (see Sec. I.A).

**FREZYB** - Geometric ratio of zone Az's in the bottom region of nonuniform zoning (see Sec. I.A).

**FREZYT** - Geometric ratio of zone Az's in the top region of nonuniform zoning (see Sec. I.A).

**GR** - Body force acceleration in the radial direction.

**GRDVEL** - Type of rezone. GRDVEL = 0. is Eulerian, = 1. is Lagrangian, and = 2. causes the rezone subroutine to be called.

**GZ** - Body force acceleration in the axial direction (usually gravity).

**GZP** - Particle acceleration in the axial direction (not applied unless particles have mass).

**IBAR** - Number of real zones in the radial direction (see Sec. I.A).

**IEOF** - Input of the record 3 section is terminated by inputting a record with IEOF = 1.

**IST** - Number of particles whose positions are to be plotted as a function of time. If IST < 0, no particles are followed.

**JBAR** - Number of real zones in the axial direction (see Sec. I.A).

**JCN** - Number of real zones from bottom of the problem to center of region of uniform zoning (see Sec. I.A).

**JUNF** - Number of zones in the axial direction in region of uniform zoning; must be an even number because JUNF/2 zones will occur above and below the point defined by JCEN (see Sec. I.A).

**KXI** - Viscosity flag (see Ref. 3).

**LAM** - Viscosity parameter (see Ref. 3).

**MU** - Viscosity parameter (see Ref. 3).

**NCLST** - Cycle number after which to terminate the run.

**NCQ** - Cycle number after which to seed the turbulence; if NCQ < 0, the turbulence is disabled; if NCQ = 0, seeding will occur based on problem time instead of cycle number, i.e. when T=TQ (see TQ).

**OM** - Relaxation parameter in the pressure iteration.

**QLEVEL** - Phenomenological turbulence viscosity parameter relating to specific turbulence energy (see Ref. 5).

**REZON** - Initial density of the ambient atmosphere at y = REZO; the density of the atmosphere above and below REZO is determined by the condition that the entire atmosphere initially be in hydrostatic equilibrium.

**REZSEE** - Specific internal energy of the entire ambient atmosphere.

**REZO** - Center of y-coordinate of the region of uniform zoning (see Sec. I.A).

**RMINEF** - Particles with RMINEF are not subject to turbulent diffusion.

**T** - Time at which the problem begins.

**TQ** - Time at which to seed the turbulence if NCQ = 0 (see NCQ).

**TSRTD** - Time at which to start turbulent particle diffusion if the turbulence is on.

**TUQI** - Proportionality constant for seeding turbulence energy. Should be chosen such that turbulence energy is a few percent of kinetic energy in any cell.

**TUSI** - Turbulence scale (constant over mesh).

(Code could be easily changed to allow scale variation throughout mesh).

**TWFIN** - Time at which to terminate the run.

**WMAXEF** - If the turbulence and particle turbulent
diffusion are on, a particle can be moved no more than $W_{MAXEF} \times \sqrt{4 \times \Sigma \Delta T}$ because of turbulent diffusion in any one cycle (see Sec. E).

$Y_B$ - Bottom of the problem mesh if the zoning is entirely uniform. If the zoning is nonuniform, $Y_B$ will be calculated internally and need not be specified.

$ZORIG$ - Number of fireball radii away from the fireball that the right problem boundary is kept.

Example:

- $\text{PARTN} DRPAR = 100., DZPAR = 100., YC = 4300., XD = 1000., YD = 0., SC = 0. $
- $\text{PARTN} DRPAR = 0. $

$T = .0083 $.

$FREZYB = 1.089359 $.

Again assuming $RESTRT = \text{.FALSE.}$, the next record will be to define marker particles.

Record 4: Namelist name - PARTN

Namelist variables

- $NAME$ TYPE (units) Possible values Default
- DRPAR Real (cm) 0. < DRPAR $\leq$ Must be specified
- DZPAR Real (cm) 0. < DZPAR $\leq$ 0.
- XC Real (cm) 0. < XC $\leq$ 0.
- XD Real (cm) 0. < XD $\leq$ Must be specified
- YC Real (cm) 0. < YC $\leq$ 0.
- YD Real (cm) 0. < YD $\leq$ 0.

(a) DRPAR - Spacing between particles in the radial direction. Particle definition cards are read until one is input with DRPAR = 0.

(b) DZPAR - Spacing between particles in the axial direction.

(c) XC - (See Sec. I.C)
(d) XD - (See Sec. I.C)
(e) YC - (See Sec. I.C)
(f) YD - (See Sec. I.C)

Example:

$\text{PARTN} DRPAR = 100., DZPAR = 100., YC = 4300., $ $XD = 1000., YD = 0., SC = 0. $ $\text{PARTN} DRPAR = 0. $

The final input will be the fireball initialization data. One namelist record is needed:

Record 5: Namelist name - FIRE

Namelist variables

- NAME TYPE (units) Possible values Default
- FBFILE Logical .TRUE., .FALSE., .FALSE.
- NRAD Integer 1 < NRAD $\leq$ 5
- NTH Integer 1 < NTH $\leq$ 180

(a) FBFILE - If .TRUE., the fireball initialization input will be found on logical unit 3.

Otherwise, the input will follow the $\text{FIRE}$ namelist card.

(b) NRAD - (See Sec. I.B).

(c) NTH - (See Sec. I.B).

Example: $\text{FIRE} FBFILE = \text{.TRUE.}$

For the form of the fireball initialization data, see the section Sec. I.B.

This completes the input for an initial setup. Restart dumps are written on each edit cycle as determined by DTO and DTQD or as specified by JDUMP and go out to logical unit 8.

To restart, a restart dump tape must be present on logical unit 7. Input record 1 must have $RESTRT = \text{.TRUE.}$. Next follows a namelist record telling from which dump to restart.

$RCYCLE INTCYC = N $ $\text{START} RESTRT = \text{.TRUE., PAPER = .FALSE., FILM = .TRUE., WRAPUP = 60.} $ $\text{RCYCLE INTCYC = -1} $ $\text{CARDN} TWFIN = 30. $
III. RESULTS OF A SAMPLE CALCULATION

A careful comparison of a YAQUI calculation with experiment was made and has been reported in Ref. 6, Fig. 1(c). In that simulation the mixed equation-of-state, turbulence, and turbulent particle diffusion options were all used. To help the user to understand the output options and to provide a comparison calculation, we include detailed results of a sample calculation, patterned after the simulation in Ref. 6.

In Sec. A the input parameters are listed along with the detailed one-dimensional fireball input data. Also given are the initial marker particle configurations, the initial grid for the complete mesh and for a smaller region surrounding the fireball, and the initial velocities.

In Sec. B contour plots of the vorticity and the specific turbulence energy at $t = .5$ s (immediately following the seeding of the turbulence) are given. Note that the general shapes of the two contour plots are similar because the turbulence energy is seeded proportional to the vorticity.

In Sec. C the positions of the marker particles and contour plots of the specific internal energy and the specific turbulence energy are given at $t = 3$ s at which time all memory of the seeding is gone. Note that the regions of greatest specific internal energy closely coincide with the regions of greatest specific turbulence energy. Also note the toroidal form of these contour plots, remembering that the left side of the mesh is an axis of cylindrical symmetry.

In Sec. D complete graphical output at $t = 10$ s is given. This is a moderately late time because torus formation occurs at $\sim t = 1.25$ s.
ALL FIREBALL INPUT DATA READ

CELL 1 12 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 2 12 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 3 12 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 4 13 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 5 14 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 6 15 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 7 16 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 8 17 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 9 18 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL
CELL 10 19 HAS A RELATIVE VOLUME ERROR OF -0.00059E+02. IT WILL BE TREATED AS A FIREBALL BOUNDARY CELL

Cycle 0, f = 0.300500E+03, Df = 1.000000E+04, cf = 2.97699E+01
GROUNDS+ 1.6503E+02, NUII+ 0. CIRC+ 3.1014X+02
DVI+ 0. IDTV+ 0. JDTY+ 0
DVT+ 0. IDTV+ 0. JDTY+ 0
TMAX+ 6.3560E+10, JTM+ 14, JMAX+ 0.
PRMT+ 1.0000E+03, PTO+ 5.3000E+03, M30+ 3.3000E+03, PAVH+ 4.3000E+03
TOTAL KINETIC ENERGY = 2.00000000E+14
TOTAL GRAV. POTENTIAL ENERGY = 4.60000000E+16
TOTAL RADIAL MOMENTUM = 1.12000000E+10
TOTAL AXIAL MOMENTUM = 2.18000000E+09
00000000000000000000000000000000000000000
VMAX = 4.000000E+04 AT VERTEX 2 17

PARTICLES
PAM= 1.920000E+03 PIB= 1.920000E+03 PTL= 2.120000E+04
YASU-LSS BALLOON = 30 x 5 - CONC. CHOS., PARTICLE FUMB. DIFF. FIX IN TANGON 10 0.300000E+03 CYCLE= 0
B. Turbulence Seeding Conditions

\[
\begin{array}{cccccc}
A & -2.50 \times 10^0 & B & -2.00 \times 10^0 & C & -1.75 \times 10^0 \\
D & -1.50 \times 10^0 & E & -1.25 \times 10^0 & F & -1.00 \times 10^0 \\
G & -7.50 \times 10^{-1} & H & -5.00 \times 10^{-1} & I & -2.50 \times 10^{-1} \\
J & 0. & K & 2.50 \times 10^{-1} & L & 5.00 \times 10^{-1} \\
M & 7.50 \times 10^{-2} & N & 3.10 \times 10^{-1} & & \\
\end{array}
\]
C. Turbulence Equilibrium Conditions

![Graph](image-url)
D. Moderately Late-Time Conditions (Eight Torus-Formation Times)

**CYCLE 15-6 FOUND**
**CYCLE 1803 FOUND**
**CYCLE 2095 FOUND**
**CYCLE 2349 FOUND**

**RESTARTING FROM CYCLE 2299**

**BALLOON - 10 x 45 - CONE. CHGS., PARTICLE TURB. DIFF., FIX IN TRBCR**

\( t = 1.000000 \times 01 \)

***** CYCLE 2299, \( t = 1.000000 \times 01 \), \( D1 = 1.21 \times 10 \times 03 \), CP = 2.97676E-07

\( QTV = 4.99933E-03 \), \( JT = 11 \)

\( QTV = 1.000000 \times 01 \), \( JT = 33 \)

\( MAX = 0.83373E-03 \), \( MAX = 0 \), \( YMAX = 0 \), \( JT = 23 \)

\( MAX = 0.83373E-03 \), \( JT = 23 \)

TOTAL INTERNAL ENERGY = 1.050576E-19

TOTAL KINETIC ENERGY = 7.139405E+12

TOTAL GRAV. POTENTIAL ENERGY = 6.666676E-16

TOTAL RADIAL MOMENTUM = -2.653527E10

TOTAL AXIAL MOMENTUM = 1.702478E10

\( VMAX = 1.12E-03 \) AT VERTEX 5 \( \times 10 \)

**PARTICLES**

\( PAR = 1.02750E+06 \), \( PAR = 0 \)

\( PART = 2.99120E+04 \)

**TAQUS = 1.955 BALLOON - 30 x 45 - CONE. CHGS., PARTICLE TURB. DIFF., FIX IN TRBCR**

\( t = 1.000000 \times 31 \) CYCLE = 3269

30
ZONES IN THE FIREBALL REGION

OMIN = 1.59333E+02  OMAX = 1.18651E+03  OMIN = 8.05204E-01
OMAX = 2.99279E+03  K = 1.95798E+04  X = 0
Y = 2.97120E+04

VACUUM BALLOON - 3D X V5 CONC. CHOS., PARTICLE TURB. DIFF., FIX IN VACUUM

1 = 1.00000E+01 CYCLE = 2299
L.00.3P0

<table>
<thead>
<tr>
<th>Order</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>6.9E+1</td>
</tr>
<tr>
<td>B</td>
<td>1.9E+2</td>
</tr>
<tr>
<td>C</td>
<td>1.8E+2</td>
</tr>
<tr>
<td>D</td>
<td>1.0E+0</td>
</tr>
<tr>
<td>E</td>
<td>2.5E-2</td>
</tr>
<tr>
<td>F</td>
<td>3.2E+0</td>
</tr>
<tr>
<td>G</td>
<td>3.1E-3</td>
</tr>
<tr>
<td>H</td>
<td>4.2E+2</td>
</tr>
<tr>
<td>I</td>
<td>5.1E+0</td>
</tr>
<tr>
<td>J</td>
<td>5.7E+0</td>
</tr>
<tr>
<td>K</td>
<td>6.4E+0</td>
</tr>
<tr>
<td>L</td>
<td>7.0E+0</td>
</tr>
<tr>
<td>M</td>
<td>7.6E+0</td>
</tr>
<tr>
<td>N</td>
<td>8.3E+0</td>
</tr>
<tr>
<td>O</td>
<td>8.9E+0</td>
</tr>
<tr>
<td>P</td>
<td>9.5E+0</td>
</tr>
<tr>
<td>Q</td>
<td>1.5E+1</td>
</tr>
<tr>
<td>R</td>
<td>1.0E+2</td>
</tr>
<tr>
<td>S</td>
<td>1.5E-3</td>
</tr>
<tr>
<td>GPM</td>
<td>1.0E+3</td>
</tr>
<tr>
<td>GPH</td>
<td>1.1E+3</td>
</tr>
</tbody>
</table>

VELOCITY MAGNITUDE
YACULITIS BALLOON - 30 X VS - CONC. CHN. PARTICLE TURN. DIFF. FIX IN TANCO

1.0E+10 Cycles: 2089
IV. FLOW DIAGRAM

Initialize

Yes

Restart

No

Read Dump File

Set Parameter Default Values

Read Parameters Overriding Values On Dump File

Read Problem Parameters

Generate Particles

Generate Mesh

Set Up Ambient Atmosphere

Set Up Fireball

Initialize For The Next Cycle

Update Vel. Explicitly Using Lagrangian Eqns.

Calculate Effects Of Turbulence

Solve Iteratively For The Time-Advanced Pressures And Velocities

Update The Energy Using Results Of Phase II

Only Lagrangian Calculation?

No

Yes

Move The Mesh With Appropriate Grid Vel.

Move The Mesh With The Fluid

Flux Mass, Momentum, And Energy

Move The Particles

Phase I, Part 1

Phase II

Phase I, Part 2

Phase III
Print Cycle Summary

Time To Seed The Turbulence?

Seed The Turbulence

Output Cycle?

Do Film Plots, Prepare Dump File, Print Hydro Variables

CP Time Exhausted, Last Cycle Reached, Termination Time Reached?

End

Yes

No

Yes

No

A

Yes

No

B
**V. CODE LISTING**

**A. LTSS-Dependent Code**

```plaintext
1  PROGRAM MAIN(TAPE1,TAPE5=TAPE1,OTAPE1,TAPE6=OTAPE1,TAPE6=108B,
2       1 TAPE3)
3   C  LTSS MAIN ROUTINE FOR YAGUI
4   C  WRITTEN BY J.L.NORTON, LASL T-3, 1975
5   C  CALL YAGUI
6   END

-----------------------------------------

1   INTEGER AND
2   FUNCTION AND(I,J)
3   C  LTSS ROUTINE TO TAKE THE BOOLEAN INTERSECTION OF TWO VARIABLES
4   C  WRITTEN BY J.L.NORTON, LASL T-3, 1975
5   C  RETURN
6   END

-----------------------------------------

1   INTEGER ANDR
2   FUNCTION ANDR(I,J)
3   C  LTSS INTERSECTION ROUTINE RETURNED IN A FLOATING-POINT VARIABLE
4   C  WRITTEN BY J.L.NORTON, LASL T-3, 1975
5   C  RETURN
6   END

-----------------------------------------

1   SUBROUTINE CLOSIT(I)
2   C  LTSS/7600 ROUTINE TO CLOSE A DISK FILE
3   C  WRITTEN BY J.L.NORTON, LASL T-3, 1975
4   CALL TPGEN(I,ITP)
```
CALL ASSIGN(I,0,ITP,-2)
RETURN
END

INTEGER COMP
FUNCTION COMP(I)
C
LTSS ROUTINE TO TAKE THE COMPLEMENT OF A VARIABLE
WRITTEN BY J.L.NORTON, LASL T=3.1975
COMP=.COMP.I
RETURN
END

INTEGER COMPR
FUNCTION COMPR(I)
C
LTSS COMPLEMENT ROUTINE RETURNED IN A FLOATING-POINT VARIABLE
WRITTEN BY J.L.NORTON, LASL T=3.1975
COMPR=.COMPR.I
RETURN
END

SUBROUTINE ECPO(SCARR, IADDLC, NW, IERROR)
C
LTSS/76/9 ROUTINE TO COPY DATA FROM LARGE CORE TO SMALL CORE
SCARR = SMALL CORE ARRAY INTO WHICH DATA IS TO BE COPIED
IADDLC = LARGE CORE ADDRESS FROM WHICH DATA IS TO BE COPIED
NW = NO. OF WORDS TO BE COPIED
IERROR = ERROR FLAG (DUMMY)
WRITTEN BY J.L.NORTON, LASL T=3.1975
LCM FWLPMC
COMMON/FWLCMC/AA1(1)
!ERROR=0
15 CALL BLOCKCOPY(AA1(IADDLC+2),SCARR,NW)
16 RETURN
17 END

-------------------------------------------------------------------

SUBROUTINE ECII(SCARR,IADDLC,NW,ERROR)

!LITSS/7608 ROUTINE TO COPY DATA FROM SMALL CORE TO LARGE CORE

SCARR = SMALL CORE ARRAY FROM WHICH DATA IS TO BE COPIED
IADDLC = LARGE CORE ADDRESS INTO WHICH DATA IS TO BE COPIED
NW = NO. OF WORDS TO BE COPIED
ERROR = ERROR FLAG (DUMMY)

WRITTEN BY J.L.NORTON, LASL T=3, 1975

COMMON/FWLCMC,AA1(I)
ERROR=#
CALL BLOCKCOPY(SCARR,AA1(IADDLC+2),NW)
RETURN
END

-------------------------------------------------------------------

INTEGER GETIT
FUNCTION GETIT(I)

!LITSS ROUTINE TO ALLOW FETCHING CONTENTS OF A WORD GIVEN ITS ABSOLUTE ADDRESS

WRITTEN BY J.L.NORTON, LASL T=3, 1975

ABSOLUTE IADDR(I)
DIMENSION IADDR(I)
GETIT=IADDR(I)
RETURN
END

-------------------------------------------------------------------

SUBROUTINE GETJOB(JOBID)


SUBROUTINE GETJTL(TL)
C
LTSS ROUTINE FOR RETURNING THE JOB TIME LIMIT IN SECONDS
C
WRITTEN BY J. L. NORTON, LASL T-3, 1975
C
CALL OATIM(I, J, K, L)
TL=FLOAT(J)*1.E-6
RETURN
END

SUBROUTINE GETLCM(IFLLCM)
C
LTSS/7600 ROUTINE TO RETURN THE AMOUNT OF LCM AVAILABLE TO THE USER
C
WRITTEN BY J. L. NORTON, LASL T-3, 1975
C
COMMON/GOBCOM/IDUM(1)
IFLLCM=IDUM(15)-IDUM(15)
RETURN
END

SUBROUTINE GETTPE(IDUM)
C
LTSS ROUTINE TO DUMMY UP TAPE FETCHING AVAILABLE ON CROS/7600
C
WRITTEN BY J. L. NORTON, LASL T-3, 1975
C
RETURN
END
SUBROUTINE LCt3UFF(FWA,NWRDS,IFILE,IFLAG,IRET,IERROR)
LISS/7600 ROUTINE TO READ OR WRITE LCM FROM OR TO A DISK FILE

FWA = FIRST LCM ADDRESS
NWRDS = NO. OF WORDS TO TRANSFER
IFILE = LOGICAL UNIT NO. OF DISK FILE
IFLAG = READ OR WRITE FLAG
  0 = READ DISK
  1 = WRITE DISK
IRET = RETURN FLAG
  0 = RETURN IMMEDIATELY AFTER ISSUING THE I/O REQUEST
  1 = WAIT UNTIL I/O IS COMPLETED BEFORE RETURNING
IERROR = ERROR FLAG
  0 = NO ERROR
  1 = ERROR
  -1 = END OF FILE ON INPUT
WITTEN BY J.L.NORTON, LASL T-3, 1975
LCM F?LC?C
COMMON/FWL?MC/AA1(1)
INTEGER FWA
CLEAR ERROR FLAG
IERROR=0
SEE WHETHER REQUEST IS READ INTO OR WRITE FROM LCM
IF(IFLAG.NE.0) GO TO 10
REQUEST IS WRITE LCM (READ DISK)
BUFFER IN(IFILE,1)(AA1(FWA+2),AA1(FWA+NWRDS+1))
GO TO 20
REQUEST IS READ LCM (WRITE DISK)
10 CONTINUE
BUFFER OUT(IFILE,1)(AA1(FWA+2),AA1(FWA+NWRDS+1))
SEE IF USER WISHES TO WAIT UNTIL I/O IS COMPLETE
20 CONTINUE
IF(IRET.EQ.0) RETURN
YES, WAIT FOR I/O TO COMPLETE
30 IF(UNIT,IFILE) 30,40,50,60
I/O SUCCESSFULLY COMPLETED. ALL DONE.
40 CONTINUE
55      RETURN
56      C      END-OF-FILE, SET ERROR FLAG AND RETURN.
57      C
58      C
59      50 CONTINUE
60      ERROR=1
61      RETURN
62      C      I/O ERROR, SET ERROR FLAG AND RETURN.
63      C
64      C
65      60 CONTINUE
66      ERROR=1
67      RETURN
68      END

------------------------------------------------------------------------

FUNCTION LCDF(I)
1      C      LTSS ROUTINE TO HANDLE LOCATION FUNCTION
2      C
3      C      WRITTEN BY J.L.NORTON, LASL T=3,1975
4      C
5      C
6      C
7      LCDF=LOC,I
8      RETURN
9      END

------------------------------------------------------------------------

SUBROUTINE NCODE(NC,IFORM,INTAB,NIN,OUTTAB)
1      C      LTSS ROUTINE TO SIMULATE CDC ENCODE STATEMENT
2      C
3      C      ENCODE(NC,IFORM,OUTTAB) (INTAB(I),I=1,NIN)
4      C
5      C      WRITTEN BY J.L.NORTON, LASL T=3,1975
6      C
7      C
8      DIMENSION INTAB(NIN),OUTTAB(1),ITEMP(10),IFORM(1)
9      DO 10 I=1,10
10     ITEMP(I)=IFORM(I)
11     REWIND 63
12     IF(MOD(NC,10),EQ,0) NW=NM+1
13     WRITE(63,ITEMP(1),I=1,NIN)
14     READ(63,20)(OUTTAB(I),I=1,NW)
15     RETURN
16     C
17     20 FORMAT(15A10)
18     END
SUBROUTINE OPENIT(IFILE,MODE)

LTSS/7600 ROUTINE TO OPEN A FILE

IFILE = LOGICAL UNIT NO. OF THE FILE
MODE = TYPE OF FILE
= 0 = BCD
= I = BINARY

WRITTEN BY J.L.NORTON, LASL T=3,1975

IF(MODE, EQ, 0) RETURN
LENGTH=1000000
IOC=0
CALL TPGEN(IFILE,ITP)
10 CONTINUE
CALL CREATE(ITP,LENGTH,IOC)
IF(IOC, GE, 0) GO TO 20
LENGTH=FLOAT(LENGTH)*,9
GO TO 10
20 CONTINUE
CALL ASSIGN(IFILE,IOC,ITP)
RETURN
END

INTEGER OR
FUNCTION OR(I,J)

LTSS ROUTINE TO TAKE THE BOOLEAN UNION OF TWO VARIABLES
WRITTEN BY J.L.NORTON, LASL T=3,1975
OR=I,UN,J
RETURN
END

INTEGER ORR
FUNCTION ORR(I,J)

LTSS FUNCTION TO RETURN THE UNION IN A FLOATING POINT VARIABLE
WRITTEN BY J.L.NORTON, LASL T=3,1975
FUNCTION RNUMF(X)
FUNCTION RNUMF(X)
C LTSS/7600 ROUTINE TO RETURN RANDOM NDS, UNIFORMLY DISTRIBUTED
C LTSS/7600 ROUTINE TO RETURN RANDOM NDS, UNIFORMLY DISTRIBUTED
ON THE INTERVAL (0,1)
ON THE INTERVAL (0,1)
C WRITTEN BY J.L.NORTON, LASL T=3,1975
C WRITTEN BY J.L.NORTON, LASL T=3,1975
C RNUMF=RNFL(X)
C RNUMF=RNFL(X)
RETURN
RETURN
END
END

SUBROUTINE SCBUFF(FWA, NWROS, IFILE, IFLAG, IRET, IERROR)
SUBROUTINE SCBUFF(FWA, NWROS, IFILE, IFLAG, IRET, IERROR)
C LTSS/760 ROUTINE TO READ OR WRITE SCM FROM OR TO A DISK FILE
C LTSS/760 ROUTINE TO READ OR WRITE SCM FROM OR TO A DISK FILE
C SEE LCBUFF FOR ARGUMENT DOCUMENTATION
C SEE LCBUFF FOR ARGUMENT DOCUMENTATION
C FWA - BEGINNING OF SCM BLOCK TO BE WRITTEN
C FWA - BEGINNING OF SCM BLOCK TO BE WRITTEN
C WRITTEN BY J.L.NORTON, LASL T=3,1975
C WRITTEN BY J.L.NORTON, LASL T=3,1975
C DIMENSION FWA(1)
C DIMENSION FWA(1)
LCM FWLCMC
LCM FWLCMC
COMMON/FWLCMC/AA(1)
COMMON/FWLCMC/AA(1)
COMMON/LCSCRC/ILSIZE, IFWASC
COMMON/LCSCRC/ILSIZE, IFWASC
C CLEAR ERROR FLAG
C CLEAR ERROR FLAG
C IERROR=0
C IERROR=0
C SEE IF THE LCM SCRATCH AREA IS LARGE ENOUGH TO HOLD THE
C SEE IF THE LCM SCRATCH AREA IS LARGE ENOUGH TO HOLD THE
SCM BLOCK
SCM BLOCK
IF(NWROS.GT.ILSIZE) CALL UNCLE(4,6HSCBUFF,25,
IF(NWROS.GT.ILSIZE) CALL UNCLE(4,6HSCBUFF,25,
125HNOT ENOUGH LCM FOR BUFFER)
125HNOT ENOUGH LCM FOR BUFFER)
C YES, SEE IF DISK IS TO BE READ OR WRITTEN.
C YES, SEE IF DISK IS TO BE READ OR WRITTEN.
C IF(IFLAG.NE.0) GO TO 6A
C IF(IFLAG.NE.0) GO TO 6A
READ DISK INTO THE LCM SCRATCH AREA

BUFFER IN(IFILE,1)(AA1(IFWASC+2),AA1(IFWASC+NWRDS+1))

WAIT FOR I/O TO COMPLETE

IF(UNIT(IFILE) EQ 10,29,30,40)

CONTINUE

COPY LCM INTO SCM

CALL BLOCKCOPY(AA1(IFWASC+2),FWA,NWRDS)

RETURN

EOF ENCOUNTERED, SET THE ERROR FLAG.

CONTINUE

ERROR=1

RETURN

DISK ERROR OR INPUT RECORD WAS SHORTER THAN EXPECTED

CONTINUE

GET THE LENGTH OF THE RECORD AND IF NOT ZERO, COPY IT TO SCM

L=LENGTH(IFILE)

IF(L.NE.0) CALL BLOCKCOPY(AA1(IFWASC+2),FWA,L)

SET THE ERROR FLAG

CONTINUE

ERROR=1

RETURN

DISK IS TO BE WRITTEN

CONTINUE

FIRST COPY SCM BLOCK TO LCM SCRATCH AREA

CALL BLOCKCOPY(FWA,AA1(IFWASC+2),NWRDS)

WRITE LCM TO DISK

BUFFER OUT(IFILE,1)(AA1(IFWASC+2),AA1(IFWASC+NWRDS+1))

WAIT FOR I/O TO COMPLETE IF SO REQUESTED

IF(IRET.EQ.0) RETURN

70 IF(UNIT(IFILE) EQ 70,80,30,50

CONTINUE

RETURN

END
SUBROUTINE SDG(X)
C LISS/76@ Routine to find sines and cosines
C of arguments in degrees
C WRITTEN BY J.L. NORTON, LASL T=3, 1975
C
DATA DEG/RAD, ©1745392519943/
Y=X*DEGRAD
SDG=SIN(Y)
RETURN
ENTRY CDG(Y)
Y=X*DEGRAD
SDG=COS(Y)
RETURN
END

INTEGER SHIFT
FUNCTION SHIFT(Ix, N)
C LISS Routine to perform bit shifting
C IX is the quantity to be shifted
C N is the no. of bits to shift, N positive means left end-around
C shift and N negative means right end-off shift.
C WRITTEN BY J.L. NORTON, LASL T=3, 1975
C
IF(N.GE.0) GO TO 10
NP=-N
SHIFT=IX, SHR, NP
RETURN
10 CONTINUE
SHIFT=IX, SHL, N
RETURN
END

SUBROUTINE STORIT(IVAR, IADD)
C LISS Routine to allow storing into a word given its absolute
C address
SUBROUTINE SYINIT

LTSS/7600 ROUTINE TO DO SYSTEM-DEPENDENT INITIALIZATION

WRITTEN BY J.L. NOR70N, LASL T=3, 1975

COMMON/FWC, YLC1, YLC2, YLC3, FILMLB, LCMSR
COMMON/FWLCMC, FWLCM
COMMON/YLC1/AA1(72300), YLC2/AA2(5000)
COMMON/YLC3/PAXY(1,200)/FILMLB/FILMBUF(4000)
COMMON/LCMSRSCR/SCRTH(1000)

* ----- BEGIN COMDECK PARAM -----*
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,

* ----- END COMDECK PARAM -----*
COMMON/LCMSR/ICSIZE, IFWASC

CHANGE THE DROPFILE NAME TO YAUQUIB

CALL CHANGE(YAUQUIB)

INITIALIZE THE FILM ROUTINES. THE ARGUMENTS ARE AS FOLLOWS --

FOR IDENTFLM -

(1) LOGICAL UNIT NO. OF THE FILM FILE
(2) ARRAY IN LCM TO BE USED AS A FILM BUFFER
(3) SIZE OF THE FILM BUFFER
(4) SIZE OF DISK FILE FOR FILM
(5) TYPE OF MICROFILM OUTPUT
   = 35 FOR 35 MM
   = 105 FOR MICROFICHE
(6) NOT USED

FOR HEADFLM -

(1) LOGICAL UNIT NO. OF THE FILM FILE
(2) NO. OF CHARACTERS IN THE HEADER (60 MAXIMUM)
(3) THE HEADER ARRAY
(4) FILM IDENTIFICATION ARRAY

FOR KEEPFLM -

(1) LOGICAL UNIT NO. OF THE FILM FILE NOT TO BE GIVEN
   TO THE SYSTEM

CALL IDENTFLM(12, FILMBUF, FLMSZ, 1000000, 105, 105)

CALL HEADFLM(12, 10, 105, 3JL4, YAUQUI, 1000000, 105)

40 C
41 C
42 C
43 C
CALL KEEPFILM(12)
C INITIALIZE THE SIZE OF THE LCM SCRATCH AREA
C ILSIZE=NLCP4
C SFT THE ADDRESS OF THE FIRST WORD OF THE SCRATCH AREA
C IFMASC=NLOC,SCRATCH(1)=LOC,FWLCH
RETURN
END

----------------------------------------------

SUBROUTINE TPGEN(IFILE,ITP)
C LTSS/7600 ROUTINE TO TAKE AN INTEGER LOGICAL UNIT NO., IFILE
C AND CONVERT IT INTO A DISK FILE NAME OF THE FORM TAPEN
C OR TAPEN, WHICH IS RETURNED LEFT-JUSTIFIED IN ITP
C WRITTEN BY J.L.NORTON, LASL T=3, 1975
C
C INTEGER OR, SHIFT, AND
C DIMENSION INUM(10)
C DATA INUM/IH0, IH1, IH2, IH3, IH4, IH5, IH6, IH7, IH8, IH9/
C C INITIALIZE
C IC=24
ITP=4HTAPE
MASk=77777777777
C SFT IF LOGICAL UNIT NO. IS ONE OR TWO DIGITS
C IF(IFILEP.LE.9) GO TO 10
C TWO DIGITS, ISOLATE THE FIRST DIGIT,
IT=IFILEP/IH0+1
IF(MOD(IFILEP,10),LE.9) IT=IT-1
C PUT THE FIRST DIGIT INTO PLACE
ITP=OR(AND(SHIFT(INUM(IT),IC),MASk),ITP)
C MODIFY PARAMETERS FOR SECOND DIGIT
IC=IC+6
MASk=77777777777
IF(IFILEP=IFILEP=10*(IT+1)
CONTINUE
SUBROUTINE TRAP(IARG)

LTSS ROUTINE TO HANDLE ERROR INTERCEPTION

WRITTEN BY J. L. NORTON, LASL 1-3, 1975

COMMON/YSCS/RESTRT, FILM, PAPER, IPO, IFD
COMMON/OVERR/IERR(121)
COMMON/GOBCOM/IG(1)
ABSOLUTE TWO(2)
INTEGER TWO, SHIFT
LOGICAL FILM
COMMON/IEDMP/IDUMP(16)

SET UP ABSOLUTE LOCATION 2 WITH A JUMP TO STATEMENT IRETA
AND SET THE EXCHANGE PACKAGE DUMP AREA TO IDUMP

TWO=2, SHL(54, UN(LOC, IRETA, SHL(30, UN(LOC, IDUMP))

SET UP ORDERLIB ERROR INTERCEPTION

DO 10 I=1, 99
10 CALL CONTROL(I, IORDL, IERO)
RETURN

ORDERLIB ERROR HAS OCCURRED, LET THE USER KNOW,

DO 20 IPX=6, IFD, 6
20 WRITE(IPX, 50)(IERR(I), I=117, 119)
IDUMP(1)=SHIFT(IERR(118), 36)
CALL PMPPK
IFLS=IG(15)
IF(FILM) GO TO 30
GO TO 40

PROGRAM HAS ABORTED

IRETA CONTINUE

DUMP THE EXCHANGE PACKAGE AND SMALL CORE ON EITHER SIDE
OF THE LOCATION OF THE ERROR

CALL PABORT

DUMP ALL OF SMALL CORE TO FILM IF FILM IS ENABLED
C IF(.NOT.,FILH) GO TO 40

C PICK THE SMALL CORE FIELD LENGTH OUT OF THE EXCHANGE PACKAGE
C IFLS=(IDUMP(3),INT((.COMP,7777777777B)),SHR,36
C CONTINUE
C IFLS=IFLS=1
C CALL DMP(0,IFLS,12)
C CONTINUE
C TERMINATE BUT SAVE THE DROPFILE
C CALL EXIT(2)
C RETURN
C
C 50 FORMAT(/IH,14MORDERLIB ERROR,13,2IH OCCURRED AT LOCATION,07,
C 129H WHICH IS THE CALL TO ROUTINE,A10)
C END

------------------------------------------------------------------------

SUBROUTINE TTYPST(IFLAG)

C LTSS ROUTINE TO SEE IF DROPFILE VARIABLE HAS BEEN CHANGED TO
C SIGNAL JOB TERMINATION
C WRITTEN BY J.L. NORTON, LASL T=3, 1975
C COMMON/OFDC/ISWITCH
C DATA ISWITCH=-1/
C IFLAG=0
C IFLAG=1
C RETURN
C END

------------------------------------------------------------------------
PROGRAM MAIN(INP, OUT, FILM, FSET7, FSET8, FSET5*INP, FSET6*OUT, FSET12*FILM, FSET3, FSET59*OUT)

CROS/7600 MAIN ROUTINE FOR YAQUI
WRITTEN BY J.L. NORTON, LASL T=3, 1975
CALL YAQUI
END

SUBROUTINE CLOSIT(I)
CROS/7600 ROUTINE FOR DESTROYING DISK FILES
WRITTEN BY J.L. NORTON, LASL T=3, 1975
CALL AFSREL(I, 0, 0, 0)
RETURN
END

SUBROUTINE GETJOB(JOBID)
CROS/7600 ROUTINE FOR RETURNING THE JOB ID
WRITTEN BY J.L. NORTON, LASL T=3, 1975
CALL GETQ4LM,JOBID)
RETURN
END

SUBROUTINE GETJTL(TL)
CROS/7600 ROUTINE TO RETURN THE JOB TIME LIMIT IN SECONDS
CALL GETQ4LMTLM,II)
TL=27.5-9*FLOAT(II)
RETURN
END
SUBROUTINE GETLCM(ISIZE)

CROS/7600 ROUTINE TO RETURN THE AMOUNT OF LARGE CORE MEMORY
AVAILABLE TO THE JOB FOR DATA STORAGE

WRITTEN BY J.L. NORTON, LASL T=3, 1975

CALL GETQ(3LKCMA, ISIZE)
RETURN

END

SUBROUTINE GETTPE(TAPE)

CROS/7600 ROUTINE FOR CHECKING TAPE LABEL AND STAGING TAPE

WRITTEN BY J.L. NORTON, LASL T=3, 1975

INTEGER TAPE, AND

TAPE WAS READ, SEE IF IT WAS LEGAL,

IF(AND(TAPE, 777777000000000000), NE, 3LXX0) CALL UNCLE(4, 6HGETTPE
2, 20HTAPE NO, INPUT ERROR)

GO TO STAGE IN THE TAPE

CALL STAGE(Y, TAPE, IFLAG)

SEE IF STAGE WAS SUCCESSFUL

IF(IFLAG, NE, 9) CALL UNCLE(4, 6HGETTPE, 23,
1 23HUNSUCCESSFUL TAPE STAGE)

YES, ALL DONE,

RETURN

END

SUBROUTINE OPENIT(IFILE, MODE)

CROS/7600 ROUTINE TO OPEN A FILE
SUBROUTINE STAGE(IF,LABEL,IFLAG)

ROUTINE TO STAGE IN A TAPE WITH LABEL=XXNNNNNN INTO FILESET IF

WRITTEN BY J. L. NORTON, LASL T=3, 1973

COMMON/YSC5/RESTRT,FILM,PAPER,IPD,IFD
DIMENSION IMDISP(5)
DATA IMDISP/4,0,0,0,0/

IFLAG IS RETURNED ZERO IF THE TAPE WAS SUCCESSFULLY STAGED, IT IS
RETURNED ONE IF FOUR STAGE ATTEMPTS FAILED.

IFLAG=0
OPEN THE FILE TO A LARGE TRACK SECTOR LIMIT
CALL OPEN(IF,R,R,R,0,0,10000,0,0)
ZEROUT THE LAST TWO CHARACTERS OF THE TAPE LABEL
IMDISP(5)=LABEL, AND (,NOT,77778)
INITIALIZE THE NO. OF STAGE ATTEMPTS
ICNT=0
GET JOB CLASSIFICATION
CALL GETQ(41,KCLA,JCLASS)
KCLASS=1LG
IF(KCLASS,NE,5) KCLASS=1LU
INITIATE THE STAGE
10 CALL CREATE(IF,KCLASS,2LST,0,0,0,IMDISP,0,0,10000,0)
INCREMENT THE STAGE COUNT
ICNT=ICNT+1
SEE IF THERE WERE ANY PARITY ERRORS

CALL PARITY(IF, ICHECK)
IF(ICHECK .NE. 0) GO TO 30

NO, PRINT MESSAGE AND RETURN.

DO 20 IPX = IDP, IFD, 6
20 WRITE(IPX, 50) LABEL, IF, IF
RETURN
30 CONTINUE

YES, SEE IF THIS WAS THE FOURTH STAGE.

IF(JCNT, LT, 4) GO TO 40

YES, SET THE ERROR FLAG AND RETURN.

IFLAG = 1
RETURN

CONTINUE

NO, RELEASE THE FILE AND TRY AGAIN.

CALL AFISREL(IF, 0, 0, 0)
GO TO 19

50 FORMAT(1HR,5HTAPE ,AB,35H HAS BEEN SUCCESSFULLY STAGED INTO ,
18HFILESET ,12H (,AB,1M))

SUBROUTINE SYINIT
  CROS/7600 ROUTINE TO PERFORM YAQUI SYSTEM INITIALIZATION
  WRITTEN BY J.L. NORTON, LASL T=3, 1975

CALL SETQ(4LKDPX, 0)
CALL OPEN(3LOUT, 0, 0, A, 9, 0, 0, 10000, 0, 0)
CALL OPEN(4LFILM, 0, 0, 0, 0, 0, 10000, 0, 0)
CALL OPEN(7, 0, 0, 0, 0, 10000, 0, 0)
CALL OPEN(3, 0, 0, 0, 0, 10000, 0, 0)
CALL MEMREQ(400000, 1)
RETURN
END
SUBROUTINE TRAP(RCOVER)

CROS/S60 routine to initialize for recovery from fatal execution

ERRORS

RCOVER is a subroutine to call after intercepting the error

written by J.L.,Norton, LASL 7-3, 1975

DIMENSION IDUMP(16)

COMMON/YSC5/RESTRT,FILM,PAPER,IPD,IFD

INTEGER SHIFT,AND,COMP

LOGICAL FILM

ASSIGN 1N TO IGDXIT

CALL XIT(IGDXIT)

IABORT is the no. of abort traps that have occurred

IABORT=0

RETURN

10 CONTINUE

CALL GETHPK(IDUMP)

CALL PABORT(IDUMP)

IF(.NOT.FILM) GO TO 20

IFLS=SHIFT(AND(IDUMP(4),COMP(7777777777B)),=36)

IFLS=IFLS=1

CALL DMP(0,IFLS,12)

20 CONTINUE

CODE HAS ABORTED, increment the abort count.

IABORT=IABORT+1

CALL the recovery routine

CALL RCOVER(IABORT)

RETURN

END
C. KRONOS-Dependent Code

1          PROGRAM MAIN(INPUT,TAPE5,INPUT,YOUT,TAPE6,YOUT,TAPE9,YOUT,TAPE7,
2                 TAPE8,FILM,TAPE12,FILM,TAPE3)
3          C          KRONOS/6600 MAIN ROUTINE FOR YAQUI
4          C          WRITTEN BY J.L. NORTON, LASL T-3, 1975
5          C          CALL YAQUI
6          END

-------------------------------

1          SUBROUTINE CLOSIT(I)
2          C          KRONOS/6600 ROUTINE TO CLOSE FILES
3          C          WRITTEN BY J.L. NORTON, LASL T-3, 1975
4          C          RETURN
5          END

-------------------------------

1          SUBROUTINE GETJOB(JOBID)
2          C          KRONOS/6600 ROUTINE TO RETURN THE JOB ID
3          C          WRITTEN BY J.L. NORTON, LASL T-3, 1975
4          C          CALL GETJN(JOBID)
5          RETURN
6          END

-------------------------------

1          SUBROUTINE GETJTL(TL)
2          C          KRONOS/6600 ROUTINE TO RETURN THE JOB TIME LIMIT IN SECONDS
3                 AS A FLOATING POINT NO.
4          C          WRITTEN BY J.L. NORTON, LASL T-3, 1975
5          C          CALL GETTL(ITL)
6          TL=ITL
7          RETURN

-------------------------------
SUBROUTINE GETLCM(IFLLCM)
C KRONOS/6600 ROUTINE TO RETURN THE AMOUNT OF LCM AVAILABLE TO
C THE USER
C WRITTEN BY J.L.NORTON, LASL T=3, 1975
C IFLLCM=500000
RETURN
END

SUBROUTINE GETTPF(IDUM)
C KRONOS/6600 DUMMY ROUTINE
C WRITTEN BY J.L.NORTON, LASL T=3, 1975
C RETURN
END

SUBROUTINE OPENIT(IFILE, MODE)
C KRONOS/6600 ROUTINE TO OPEN A FILE (DUMMY)
C WRITTEN BY J.L.NORTON, LASL T=3, 1975
C RETURN
END

SUBROUTINE SYINIT
C
IDENT TRAP

KRONOS/6600 ROUTINE TO INTERCEPT HARDWARE OR SOFTWARE ABORTS

* REGISTERS AND SMALL CORE ARE DUMPED TO OUTPUT, A MESSAGE IS PUT
* IN THE SYSTEM DAYFILT, AND THE EXTERNAL SUPPLIED IN THE CALL
* TO TRAP (CALL TRAP(RCOVER)) IS CALLED WHENEVER AN ERROR
* IS DETECTED

* WRITTEN BY J.L. NORTON, LASL T-3, 1975

ENTRY TRAP
TRAP DATA 0
SX6 B1
SA6 SAVA
EXIT RCOVER
JP TRAP
RCOVER SYSTEM DMP,R,0,0
SYSTEM DMP,R,135000B,0
MESSAGE MESS,3,R
SA1 SAVA
SR1 X1+1
JP B1
MESS DATA 17L*** JOB ABORT ***
DATA 0
SAVE DATA 0
END
D. KRONOS/CROS-Dependent Code

FUNCTION AND(I,J)

KRONOS/6600 AND CROS/7600 ROUTINE TO DO BOOLEAN INTERSECTION

WRITTEN BY J.L. NORTON, LASL T-3, 1975

INTEGER AND
AND=I AND J
RETURN END

FUNCTION OR(I,J)

KRONOS/6600 AND CROS/7600 ROUTINE TO BOOLEAN INTERSECTION
AND RETURN THE RESULT IN A FLOATING POINT VARIABLE

WRITTEN BY J.L. NORTON, LASL T-3, 1975

INTEGER OR
OR=I OR J
RETURN END

FUNCTION CPL(I)

KRONOS/6600 AND CROS/7600 ROUTINE TO COMPLEMENT A WORD

WRITTEN BY J.L. NORTON, LASL T-3, 1975

INTEGER CPL
CPL=NOT I
RETURN END

FUNCTION COMPP(I)

KRONOS/6600 AND CROS/7600 ROUTINE TO COMPLEMENT A WORD

WRITTEN BY J.L. NORTON, LASL T-3, 1975
IDENT GETIT

* CROS/7600 AND KRONOS/6600 FUNCTION GETIT(IADD) TO PICK UP THE
* CONTENTS OF ABSOLUTE LOCATION IADD

ENTRY GETIT
GETIT DATA 0
SA1 R1
SA2 X1
BX6 X2
JP GETIT
END

SUBROUTINE LCBUFF(F,A, NORDS, IFILE, IFLAG, IRET, IERROR)

F,A = FIRST LCM ADDRESS
NORDS = NO. OF WORDS TO TRANSFER
IFILE = LOGICAL UNIT NO. OF DISK FILE
IFLAG = READ OR WRITE FLAG
IFLAG = "R" = READ DISK
I = ARITH DISK
IPAT = RETURN IMMEDIATELY AFTER ISSUING THE I/O REQUEST
I = WAIT UNTIL I/O IS COMPLETED BEFORE RETURNING
I = NOT FUNCTIONAL ON CROS/7600 OR KRONOS/6600
IERROR = ERROR FLAG
I = NO ERROR
I = ERROR
I = END-OF-FILE ON INPUT
I = WRITTEN BY J.L. HORTON, LASL 1975
1.4 = BEGIN COBOL PARAM
1 = COMMON/PCOM/NSCP1, ITARP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
25 COMMON/IFLAG,FNAP
26 COMMON/YSC1,AASC(1)
27 INTEGER FNAP,FNAP
28 C
29 C CLEAR ERROR FLAG
30 C IERROR=1
31 C FNAP IS THE BEGINNING LCM ADDRESS OF THE PORTION OF LCM BEING
32 C READ OR WRITTEN
33 C FNAP=FNA
34 C NRDSN IS THE TOTAL NO. OF WORDS OF LCM THAT HAVE BEEN READ
35 C OR WRITTEN
36 C NRDSN=NRDSN
37 C LOOP FOR READING TO OR WRITING FROM LCM FROM OR TO DISK IN BLOCKS
38 C 10 CONTINUE
39 C NW IS THE NO. OF LCM WORDS TO TRANSFER THIS TIME. IT IS EQUAL
40 C TO THE SIZE OF THE SCM BUFFER UNLESS THE NO. OF LCM WORDS
41 C LEFT TO TRANSFER IS LESS THAN THE BUFFER SIZE. IN THE LATTER
42 C CASE, NW IS JUST SET TO THE NO. OF REMAINING WORDS.
43 C NW=NRDS1
44 C NRTEST=NROIS+NW
45 C IF(NRTEST>NRDS) NW=NRDS-NRDSN
46 C SET WHETHER REQUEST IS READ INTO OR WRITE FROM LCM
47 C IF(IFLAG,NE,1) GO TO 80
48 C REQUEST IS WRITE LCM (READ DISK)
49 C BUFFER IN(IFILE,1)(AASC(1),AASC(NW))
50 C WAIT FOR I/O TO COMPLETE
51 C 20 IF(UNIT,IFILE) 20,50,40,30
52 C ERROR OCCURRED IN DISK TRANSFER (EITHER UNEXPECTED EOF, PARITY
53 C ERROR, OR RECORD SHORTER THAN EXPECTED), SET THE ERROR FLAG
54 C AND RETURN.
55 C 30 CONTINUE
56 C IERROR=1
57 C RETURN
58 C 40 CONTINUE
59 C IERROR=1
60 C RETURN
61 C 50 CONTINUE
62 C CALL ECWR(AASC,FNAP,NK,IDUM)
R2 C BLOCK TRANSFER COMPLETED. UPDATE HARDSD AND SEE IF THERE IS
R3 C MORE DATA LEFT TO TRANSFER.
R4 C
R5 C CONTINUE
R6 N=SIZE(NHARDSD),N
R7 IF(IFLAG.NE.1) GO TO 73
R8 N=LENGTH(IFILE)
R9 IF(N.NE.N) GO TO 31
R10 CONTINUE
R11 IF(NHARDSD.NE.NHARDSD) RETURN
R12 F+AP=F+AP+N
R13 GO TO 10
R14 C REQUEST IS READ LCM (WRITE DISK)
R15 C
R16 C CONTINUE
R17 CALL FCON(AASC,F+AP,NH,IDUM)
R18 BUFFER OUT(IFILE,1)(AASC(1),AASC(NH))
R19 C WAIT FOR I/O TO COMPLETE
R20 C
R21 C IF(UNIT,IFILE) 90,60,40,30
R22 C END

SUBROUTINE CNODE(NC,IFORM,INTAB,IMK,OUTTAB)
C KRONOS/660 AND CROS/760 ROUTINE TO SIMULATE THE CDC ENCODE
C STATEMENT
C
C FNCODE(NC,IFORM,OUTTAB) (INTAB(I),IMK,NIN)
C
C WRITTEN BY J.L. NORTON, LASL T-3, 1975
C
C DIMENSION INTAB(NIN)
C INTEGER OUTTAB
C FNCODE(NC,IFORM,OUTTAB) INTAB
C RETURN
C FND

FUNCTION OR(I,J)
C KRONOS/660 AND CROS/760 ROUTINE TO DO BOOLEAN UNION
C
C WRITTEN BY J.L. NORTON, LASL T-3, 1975
7 INTEGR OR
8 OR=1, OR=I
9 RETURN
12 END

----------------------------------------
1 FUNCTION ORR(I,J)
2 C K顺着/660 和 C顺着/760 ROUTINE TO DO BOOLEAN UNION
3 C AND RETURN THE RESULT IN A FLOATING POINT VARIABLE
4 C WRITTEN BY J.L. NORTON, LASL T=3, 1975
5 C INTEGR ORR
6 OR=1, OR=I
7 RETURN
8 END

----------------------------------------
1 FUNCTION RNUMF(X)
2 C K顺着/660 和 C顺着/760 ROUTINE TO RETURN RANDOM NOS. WITH
3 C UNIFORM DISTRIBUTION ON THE INTERVAL (0,1)
4 C WRITTEN BY J.L. NORTON, LASL T=3, 1975
5 C RNUMF(RNUMF(X))
6 RETURN
7 END

----------------------------------------
1 SUBROUTINE SCPUFF(FHA, K顺着/660, IFFILE, IFLAG, IRET, IERROR)
2 C K顺着/660 AND C顺着/760 ROUTINE TO READ OR WRITE SCP FROM
3 C OR TO A DISK FILE
4 C SEE LCPUFF FOR ARGUMENT DOCUMENTATION
5 C FHA = BEGINNING OF SCP BLOCK TO BE WRITTEN
6 C WRITTEN BY J.L. NORTON, LASL T=3, 1975
7 C
DIMENSION FWA(1)
ERROR=0
IF(IFLAG.NE.4) Go To 10
BUFFER IN(IFILE,FWA(1),FWA(NWRDS))
GO TO 20
10 CONTINUE
BUFFER OUT(IFILE,FWA(1),FWA(NWRDS))
20 CONTINUE
IF(TRET.EQ.9) RETURN
30 IF(UNIT,IFILE) 30,40,50,60
40 CONTINUE
50 IF(IFLAG.NE.0) RETURN
60 N=LENGTH(IFILE)
70 IF(N.NE.NWRDS) Go To 60
RETURN
80 CONTINUE
20 IFERROR=1
RETURN
60 CONTINUE
1 ERROR=1
RETURN
END

CENT SHIFT

ENTRY SHIFT

* WRONDS/664 AND CROS/7649 ROUTINE TO REPLACE THE LAST RUN
* COMPILER IN-LINE SHIFT FUNCTION, THIS IS NECESSARY BECAUSE
* OF THE NEED TO INCLUDE SHIFT IN AN INTEGER STATEMENT FOR
* LETSS COMPATIBILITY, OTHERWISE DOING SO FORCES A CALL TO AN
* EXTERNAL FUNCTION.
* WRITTEN BY J.L. HORTON, LASL T=3, 1975

SHIFT DATA 0
17 SA1 N2
18 SA2 N1
19 ZR X1,ZERO
20 PL X1,LEFT
21 PX3 -X1
22 SH3 X3
23 AX6 R3,X2
24 JP SHIFT
25 LEFT SB3 X1
26 LX6 R3,X2
27 JP SHIFT
28 ZER0 BX0 X2
INERT STORIT

* CROS/7600 AND KRONOS/6600 SUBROUTINE STORIT(INORD, IADD) WHICH
* PUTS INORD INTO ABSOLUTE SCM LOCATION IADD
.
ENTRY STORIT

ENTRY STORIT

VFD 2/RHSTORIT,19/2
STORIT DATA 0
SA1 B1
SA2 R2
RXB X1
SA6 X2
JP SIGRIT
END

SUBROUTINE ITYIST(IFLAG)

C KRONOS/6600 AND CROS/7600 ROUTINE TO DUMMY UP CAPABILITY OF
C LTSS TO MODIFY THE DROPFILE
C WRITTEN BY J. L. NORTON, LARL T=3, 1975
IFLAG#E RETURN
END
SUBROUTINE YAQUI

YAQUI IS A TWO-DIMENSIONAL FLUID DYNAMICS CODE THAT COMBINES
THE ICE (IMPLICIT CONTINUOUS EULERIAN) AND ALE (ARBITRARY
LAGRANGIAN-EULERIAN) METHODS, ALLOWING CALCULATION OF FLOWS
AT ALL SPEEDS.

ORIGINALLY WRITTEN BY A.A. AAMSDEN, HANS RUPPEL, AND C.W. HIRT, L&SL T=3
MODIFIED AND DOCUMENTED BY J.L. NORTON, L&SL T=3, 1974

* ----- BEGIN COMECK PARAM ----- *
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCp1, NLCp2,
1 NLCp3, NLCp4, IFLMS2
* ----- END COMECK PARAM ----- *

LCM IS SET UP IN THE FOLLOWING FASHION ==
BLOCK 1, NLCp1 WORDS LONG, ARRAY DATA
BLOCK 2, NLCp2 WORDS LONG, PARTICLE POSITIONS AND MASSES
BLOCK 3, NLCp3 WORDS LONG, TIME-DEPENDENT PARTICLE POSITIONS
BLOCK 4, IFLMS2 WORDS LONG, FILE FILE BUFFER FOR LTSS
BLOCK 5, NLCp3 WORDS LONG, SCRATCH AREA

* ----- BEGIN COMECK YAQ8C ----- *
LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAH, MU
COMMON/YSC1/AABC(NSCP1)
COMMON/YSC1/AASC(9600)
COMMON/YSC2/AA(1), ANC, A0, ABFAC, ABM, B0, COLAH, CYL, DR, DT, DTC, DTPAC,
1 DTOC(10), DTOC1(10), DTO2, DTOB, DTPO, DTZ, E110, EPS, FIPXL, FIPXR,
2 FIPY8, FIPY9, FIPX, FIXR, FIXY8, FIXY9, FRXZ, GRDVEL, GZ, GZP, I, IBAR,
3 IDT, IJ, IJ, IJ, IP1, IP2, IP3, 1SC, 1SC, 1SC, 1TV,
4 INF, IX, IX, IX, IX, J, JBAR
COMMON/YSC2/QCEN, P1, P2, P3, JUNF, JUNFO2, KXI, LAH, LP8, MU, NAME(8),
1 NCYC, NLC, NPS, P, NQ1, NQ2, NC, NUm, ZORIG, OM, HCYL, PXCONV
2, PXL, PXR, PXB, PXCONV, PVT, RDT, REZON, REZIF, REZY0, RIBAR, RIBJ0,
3 FRESY, FREEY, ROMFY, T, THIRD, NCLS, TOUT, TWFIN
COMMON/YSC2/TUG1, TUG2, NC, TNEG, TNEG, TUSV, TURB, TTOP, TPRTE, PDBM,
1 TLG, NTLG, TP5, TUPOT, TUBAV, TK1, TUEENG, EPI, BAV1, QLEVEL, TQ, IBT,
2 VV, XCONV, XL, XR, Y, YCONV, VT, TPPO, DTB1, DTB2, DTHAS, FM, YBO, YCNVLD,
3 XRCONV, FIXR, FIXLX, FIX1X, IXJL, XJL, JIM, JVM, QM, QMX, WMX, JNH, T2, TLI,
4 RMRFX, RMRFY, JVM, TTHAS, JTF, DTHAS, DTASV, DTCGAV, IDTV
5 JDVC, IDTC, JDC, JDC, JDC, JDC, JDC, TOTR, UQ, Q, Q, Q, Q, Q, Q, Q, Q, Q,
6 TMAS, WMAXEP, RIME, P, TSTR, TSTR
COMMON/YSC2/ZZ
COMMON/YSC2/ITAB(1698)
COMMON/YSC2/ITABP(1698)
COMMON/YSC2/ITABXP(1698)
COMMON/YSC2/ITABYP(1698)

* ----- END COMECK YAQ8C ----- *
COMMON/FTABC/FTAB(2)
INTEGER AA1
EXTERNAL YEXIT
DATA FTA8/6,12/
DATA NSCP1/9600/, ITABP/1698/, ITABXP/1698/, ITABYP/1698/
DATA NP1/10/, NP2/20/, NLCp1/1250/, NLCp2/5000/
DATA TP/0/.
DO ANY NECESSARY SYSTEM INITIALIZATION

CALL SYINIT

INITIALIZE THE ERROR RECOVERY ROUTINE

CALL TRAP(YEXIT)

GET THE CP TIME AT JOB STARTUP

CALL SECOND(TP)

WRITE(59,38) TP

initialize

CALL YINIT

CHECK TYPE OF RUN

IF(RESTART) GO TO 10

RUN IS A NEW PROBLEM, GO GENERATE IT.

CALL YASET

GO TO 20

RUN IS A PROBLEM RESTART, READ THE DUMP TAPE.

10 CONTINUE

CALL YAR8RT

GET RID OF THE DUMP FILE

CALL CLOSIT(7)

20 CONTINUE

EXECUTE THE MAIN CODE

CALL YAQUI2

RUN TERMINATION, EXIT.

CALL UNCLE(2,SHYAQUI,15,15HRUN TERMINATION)

30 FORMAT(1H,29HBEGIN CODE EXECUTION AT CP = ,F10,4)

END

-------------------------------

SUBROUTINE AIR

C SEMI-PHYSICAL FIT TO THE EQUATION OF STATE OF AIR
DENSITIES FROM 10**2 TO 10**7 NORMAL DENSITY
PRESSURE=(GAMMA=1)*RHO*E, WHERE GAMMA IS A FUNCTION OF
DENSITY AND ENERGY
RHO=MATERIAL DENSITY
EJLN=MATERIAL SPECIFIC INTERNAL ENERGY
GMONE=GAMMA=1.
CONCJ=RELATIVE CONCENTRATION, SEE NOTE BELOW.
ALL UNITS ARE CGS FOR INPUT QUANTITIES
ORIGINALLY OBTAINED FROM THE AIR FORCE WEAPONS LAB
MODIFIED BY J.L. NORTON, LABL T=3, 1974
COMMON/EGNST/RHO,EJLN,GMONE,CONCJ

IN THIS VERSION OF THE CODE, TWO EQNS OF STATE ARE BEING USED, AIR
AND METHANE, THE FINAL VALUE OF GMONE IS DETERMINED BY
THE RELATION GMONE(FINAL)=CONCJ*GMONE(METHANE)
+(1.-CONCJ)*GMONE(AIR).

THUS, CONCJ_1, YIELDS A METHANE GAMMA AND CONCJ=0., AIR.
ANYTHING IN BETWEEN USES A LINEAR COMBINATION OF THE TWO,

CHECK TO MAKE SURE CONCJ DOES NOT EXCEED 1, IF IT DOES, SET IT TO 1
IF(CONCJ.GT.1.) CONCJ=1.
IFLAG=1 SIGNALS AIR TO DO THE GAMMA LINEAR COMBINATION, IF
CONCJ=0., THIS IS UNNECESSARY.
IFLAG=0
IF CONCJ=0., SKIP THE METHANE CALCULATION ENTIRELY
IF(CONCJ.EQ.0.) GO TO 10
GO GET THE METHANE GAMMA
CALL MTHANE
IF CONCJ=1., WE ARE ALL DONE
IF(CONCJ.EQ.1.) RETURN
TURN ON THE LINEAR COMBINATION FLAG, SAVE THE METHANE GMONE, AND
CONTINUE ON WITH THE AIR EOS CALCULATION
IFLAG=1
GMONE=GMONE
CALCULATE GAMMA=1. FOR AIR
10 CONTINUE
RHOZ IS THE NORMAL AIR DENSITY
RHOZ=1.293E=3
RHOT IS THE COMPRESSION

RHOT=RHO/RHOZ

MAKE E POSITIVE AND CONVERT TO SI UNITS (TJ/MG)

E=1.6*ABS(EJLN)

THE ENERGY AT WHICH OXYGEN AND NITROGEN DISSOCIATE IS A
FUNCTION OF DENSITY

E1=(8.5=E)/1.975

THE FERMI-DIRAC FUNCTION IS ONLY COMPUTED WITHIN 5*DELTA E OF
EACH TRANSITION, OTHERWISE IT IS ZERO OR ONE.

IF(ABS(E1)=5.150,20,20
20 IF(E1) 60,40,30
30 FON=EXP(-E/4.46)
40 FON=0.
50 GO TO 60
60 FON=EXP(-E/6.93)
70 W=0.
80 GO TO 60
90 50 DE1=.975*RHOT**.05
91 EE1=.5*397*ALOG10(RHOT)
92 E1=(EE1=E)/DE1
93 W=1./EXP(-E1+1.)
94 FON=EXP(-E/4.46)*W
95 FON=EXP(-E/6.93)*(1.-W)
96 THE DENSITY DEPENDENCE ONLY OCCURS ABOVE E=1., AND IT IS OF
97 THE FORM (RHO/RHOZ**2*CONSTANT*LOG(E)). THE CONSTANT
98 MAKES A TRANSITION FROM .048 TO .029 AS THE OXYGEN
99 DISSOCIATES AND THE DENSITY SPREAD BECOMES CONSTANT BEYOND
100 THE SECOND PEAK.

60 IF(E1) 70,70,80
70 BETA=0.
80 GO TO 90
90 80 BETA=(.348*W5+.032*W3)*ALOG10(E)
100 E2=EXP(-40.)/3.
110 IF(ABS(E2)=5.) 130,100,100
120 100 IF(E2) 110,110,120
130 110 FON=0.
140 W=0.
150 GO TO 140
160 120 FON=EXP(-E/25.5)
170 W=0.
180 GO TO 140
190 130 STOP
200 STOP
SUBROUTINE BC(IFLAG)

ROUTINE TO SET RIGID WALL BOUNDARY CONDITIONS IN YAQUI

IFLAG = INDEX INDICATING WHICH VELOCITY ARRAYS ARE TO BE SET

= 1 = UTIL,VTIL

= 2 = UL,VL

= 3 = UP,VP

WRITTEN BY J.L.NORTON, LASL T-3, 1975

----- BEGIN COMDECK YADIM -----  

DIMENSION X(1),XPAR(1),R(1),YPAR(1),Y(1),MPAR(1),UC(1),UG(1),DELSl
1 1(V(1),VG(1),RO(1),8IE(1),MP(1),RMP(1),RCBG(1),E(1),ETIL(1),RVOL
2 1(1,M(1),RM(1),VP(1),P(1),PL(1),UP(1),UTIL(1),UL(1),CO(1),VTIL(1)
3 1(VL(1),ROL(1),AVXSV(1),AVYSV(1),DLBROI(1),DLBROG(1),CAPGM(1),TUQ
4 1(S,9G(1),TUS(1),GRROR(1),GRRDZ(1),GRROP(1),TUGVEC(1),MTIL(1)
5 1(1,CONC(1),CTEMP(1),ANCU(1),ANCV(1),GRBV(1),GZBV(1),X13K(1),X24K(1)
6 1(Y13K(1),Y20K(1),X16K(1),MR29K(1),DLBMB(1),ARE(1)
7 *  END COMDECK YADIM (4)

----- BEGIN COMDECK YADIM -----  

LOGICAL RE1RT,FILM,PAPEr.TURB

REAL LAM,HU

COMMON/YSC/AA8C(N8C1)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)

COMMON/Y8C/AA8C(9880)
Common/YSC2/ITAB(IITABP)
Common/YSC4/ITAB(1000)
Common/YSC5/RESTR/FLM,PAPER,IPD,IFD

* * * * * * BEGIN COMDECK YAGSC * * *

Equivalence(AABC(1),X,XPAR),(AABC(2),R,YPAR),(AABC(3),Y,MPAR),
(AABC(4),U),(AABC(5),V),(AABC(6),RO),(AABC(7),DELSM,RCBG,MP),(AASC  
1 (8),E,ETIL,AREA,XR13K),
2 (AABC(14),BIE),(AABC(16),PM0,DKLSM,RMP),(AABC(9  
3 ),RVOL),(AABC(10),M,RH,VP),(AABC(11),P,PL,EP,UP),(AABC(12),UTIL  
4 UL,PMX,PV),(AABC(13),VTIL,VL,PMY,PV),(AABC(14),G,CO,ROL),(AABC(17  
5 ),CAPGAU,UG),(AABC(18),TU0),(AABC(19),SIG),(AABC(20),TU),(AABC(  
6 EL,GROR),(AABC(22),GROR2),(AABC(23),DLSRO,Y13K),(AABC(24),GZBV  
7 ),(AABC(25),DLSRO,V0),(AABC(26),GRBV),(AABC(27),GRROP,TUBVEC,  
8 Y24K),(AABC(28),MTIL),(AABC(29),CND),(AABC(30),CTEMP,VR24K),
9 AABC(31),ANCU),(AASC(32),ANCV),(AABC(33),AVXBV,K13K),(AABC(34),  
1 AVSVX24K)

REAL M,MP,MPAR,MTIL

* * * END COMDECK YAGSC * * *

* * * END COMDECK YSTORE * * *

SET UP THE LOOP OVER CELLS

CALL START

SET THE BOTTOM BOUNDARY VELOCITIES

IJBV=IJ
DO 40 IJ=1,IP1
GO TO C10,20,30),IFLAG

10 CONTINUE
VTIL(IJ)=0.
GO TO 40

20 CONTINUE
VL(IJ)=0.
GO TO 40

30 CONTINUE
VP(IJ)=0.
GO TO 40

40 IJ=IJ+NO
IJB=IJBV

SET THE LEFT AND RIGHT BOUNDARIES

DO 90 JEB=JP1
GO TO C50,60,70),IFLAG

50 CONTINUE
SUBROUTINE CINIT

ROUTINE TO CALCULATE QUANTITIES FOR THE CYCLE PRINT AND
INITIALIZE FOR THE NEXT CYCLE

ORIGINALLY WRITTEN BY A.A. AMSDEN, LABL T=3
COMMON/EGNST/RTMP,ETMP,GMONE,CONCJ

INITIALIZE LOOP VARIABLES

POTE=0.
TK=0.
TI=0.
UMOM=0.
VMOM=0.
TMASS=0.
TUENG=0.
CIRC=0.
TMAX=0.
TGMAX=0.
ITM=0.
ITG=0.
JTM=0.

LOOP OVER ALL REAL ZONES IN THE MESH

CALL START

DO 60 J=2,JP1

DO 50 I=1,ISAR

IPJ=IPJ+NP

IPJ=IPJ+NP

TMAX IS THE MAXIMUM SIZE OF ANY CELL IN THE MESH, IT IS FOUND IN CELL ITM,JTM.

IF(SIE(IJ),LE,TMAX) Go To 10

ITM=I

JTM=J

TMAX=SIE(IJ)

TGMAX IS THE MAXIMUM SIZE GRADIENT AND (ITG,JTG) IS THE CELL IN WHICH IT OCCURS

10 CONTINUE

SAVE=X(IPJ)-X(IJ)**2+(Y(IPJ)-Y(IJ))**2

SAVE=X(IPJ)-X(IJ)**2+(Y(IPJ)-Y(IJ))**2

SAVE=ABS(SIE(IJ)*SIE(IPJ))/SQRT(SAVE)

SAVE=ABS(SIE(IJ)*SIE(IPJ))/SQRT(SAVE)

SAVE=MAX(SAVE,SAVE)

IF(SAVE.LT,TGMAX) GO TO 20

ITG=I

JTG=J

TGMAX=SAVE

20 CONTINUE

CALCULATE THE CIRCULATION AROUND THE PROBLEM BOUNDARIES

IF(I.EQ.1) CIRC=CIRC+0.5*(Y(IJP)-Y(IJ))*(X(IPJ)-X(IJ))

IF(I.EQ.1) CIRC=CIRC+0.5*(V(IJP)-V(IJ))*(Y(IPJ)-Y(IJ))

IF(J.EQ.3) CIRC=CIRC+0.5*(U(IJP)-U(IJ))*(X(IPJ)-X(IJ))
IF(J.EQ.JBAR) CIRC*CIRC*0.5*(U(IJ)*U(IPJ))*(X(IPJ)-X(IJ))

XMSENG IS THE MASS IN CELL I,J
THASS IS THE TOTAL PROBLEM MASS

XMSENG*PR(IJ)/RVOL(IJ)
THASS*TMASS*XMSENG

SPENGK IS THE SPECIFIC KINETIC ENERGY OF CELL I,J

SPENGK=0.125*(U(IPJ)**2+U(IPJP)**2+U(IJP)**2+V(IPJ)**2+V(IPJP)**2+V(IJP)**2)

TK IS THE TOTAL KINETIC ENERGY IN THE PROBLEM

TK=TK*SPENGK*XMSENG

TI IS THE TOTAL INTERNAL ENERGY IN THE PROBLEM

TI=TI*XMSENG*SIE(IJ)

UMOM AND VMOM ARE THE RADIAL AND AXIAL TOTAL MOMENTA, RESPECTIVELY

UMOM=UMOM+0.25*XMSENG*(U(IPJ)+U(IPJP)+U(IJP)+U(IJ))
VMOM=VMOM+0.25*XMSENG*(V(IPJ)+V(IPJP)+V(IJP)+V(IJ))

POTE IS THE TOTAL GRAVITATIONAL POTENTIAL ENERGY (WITH GM1).
NOTE THAT VERTEX Y#S AND MASSES ARE USED.

POTE=POTE+Y(IJ)/RM(IJ)

ADD THE J=JP2 VERTEX INTO THE POTENTIAL ENERGY

IF(J.EQ.JP1) POTE=POTE+V(IPJ)/RM(IJ)
IF(TURB) TUQENG=TUQENG*TUQ(IJ)*XMSENG

NEW PRESSURE MUST BE CALCULATED, P ARRAY CURRENTLY HOLDS VELOCITIES DUE TO EQUIVALENCING OF P WITH UP.

PUT A LOWER BOUND OF ZERO ON THE SIE

SIE=AMAX1(SIE(IJ),0)

GO GET THE CELL GAMMA=1

ROTM=ROT(IJ)
ETMP=SIET
CONCJ=CONC(IJ)
CALL AIR
GM1=GMONE

COMPUTE PRESSURE

P(IJ)=GM1*ROTM*SIET

SET PRESSURE OF BOTTOM FICTIONIOUS ROW USING GAMMA OF CELLS ABOVE
178 C IF(J,EQ,2) P(IJM)=GMI*RO(IJM)*SIE(IJM)
179 C SET PRESSURE OF TOP FICTITIOUS ROW USING GAMMA OF CELLS BELOW
180 C IF(J,EQ,JP1) P(IJP)=GMI*RO(IJP)*SIE(IJP)
181 C SEE IF WE ARE PROCESSING THE RIGHTMOST REAL COLUMN
182 C IF(I,NE,IBAR) GO TO 40
183 C YES,
184 C SET PRESSURE OF RIGHT FICTITIOUS COLUMN USING GAMMA OF CELLS
185 C TO THE LEFT
186 C P(IJP)=GMI*RO(IJP)*SIE(IJP)
187 C SEE IF WE ARE PROCESSING THE BOTTOM REAL ROW
188 C IF(J,NE,2) GO TO 30
189 C YES,
190 C SET PRESSURE OF BOTTOM CELL IN RIGHT FICTITIOUS COLUMN USING
191 C GAMMA OF CELL (IBAR,2)
192 C IFJM=IJM+NO
193 C P(IJM)=GMI*RO(IJM)*SIE(IJM)
194 C GO TO 40
195 C 30 CONTINUE
196 C NO, SEE IF WE ARE PROCESSING THE TOP REAL ROW.
197 C IF(J,NE,JP1) GO TO 40
198 C YES,
199 C SET PRESSURE OF UPPER CELL IN RIGHT FICTITIOUS COLUMN USING
200 C GAMMA OF CELL (IBAR,JP1)
201 C P(IJP)=GMI*RO(IJP)*SIE(IJP)
202 C 40 CONTINUE
203 C IJP=IPJ
204 C IJM=IJM+NO
205 C IJM=IJM+NO
206 C 50 CONTINUE
207 C ADD THE IP1 VERTEX INTO THE POTENTIAL ENERGY
208 C POTEPOTEPY(IJP)/RMI(IJP)
209 C ADD THE (IP1,JP2) VERTEX INTO THE POTENTIAL ENERGY
210 C IF(J,EQ,JP1) POTEPOTEPY(IJP)/RMI(IJP)
211 C CALL LOOP
212 C 60 CONTINUE
SUBROUTINE CONTOUR(ILOG, ITITLE, NWT)

C ROUTINE FOR DOING CONTOUR PLOTS IN YAQUI

C ILOG IS 0 IF LINEAR CONTOUR INCREMENTS ARE TO BE USED
C ILOG IS 1 FOR LOGARITHMIC CONTOUR INCREMENTS
C ITITLE IS THE HOLLERITH TITLE TO BE USED ON THE PLOT
C NWT IS THE NO. OF COMPUTER WORDS IN THE TITLE

C THE QUANTITY TO BE PLOTTED IS FOUND IN CQ
C
C ORIGINALLY WRITTEN BY A.A. AMSDEN, LASL T=3
C MODIFIED AND DOCUMENTED BY J.L. NORTON, LASL T=3, 1975

C ----- BEGIN COMDECK PARAM -----
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFILMB
C ----- END COMDECK PARAM -----

* ----- BEGIN COMDECK YADDIM -----
DIMENSION X(1), XPAR(1), Y(1), YPAR(1), U(1), UG(1), DELSM(1),
1 V(1), W(1), RO(1), ROEL(1), MP(1), RMSQ(1), ETIL(1), RVOL(1),
2 H(1), RH(1), UP(1), PL(1), UL(1), CQ(1), VTIL(1),
3 VL(1), RO(1), AXS(1), AXYS(1), AVL(1), DLBSQ(1), DLBSQ(1), CAPGAK(1),
4 SIG(1), TUS(1), ERR(1), ERR(1), ERR(1), ERR(1), TUGVEC(1), NTLT(1),
5 CONC(1), CTMP(1), ANCC(1), ANCV(1), GRV(1), OZSV(1), OZSV(1),
6 Y13K(1), Y24K(1), X13K(1), X24K(1),
7 DKLSM(1), AREA(1)

* ----- END COMDECK YADDIM -----

* ----- BEGIN COMDECK YASC -----
LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAM, MU
COMMON/VSC1/AASC(NSCP1),
 COMMON/VSC1/AASC(NSCP2),
 COMMON/VSC2/AAC(NSCP1),
 COMMON/VSC2/AAC(NSCP2),
 COMMON/VSC2/JCN, JP1, JP2, JP3, JUNF, JUNC, KXI, LAM, LBP, MU, NAME(8),
 NCY, NLC, NP3, NP1, NP2, NP3, NQ, NQ2, NQ3, NQ4, NQ5, NQ5, NQ6, NQ7,
 NL1, NL2, NL3, NL4, NL5, NL6, NL7, NL8, NL9, NL10, NL11, NL12, NL13,
 PR1, PR2, PR3, PR4, PR5, PR6, PR7, PR8, PR9, PR10, PR11, PR12, PR13,
 REF1, REF2, REF3, REF4, REF5, REF6, REF7, REF8, REF9, REF10, REF11,
 REF12, REF13, REF14, REF15, REF16, REF17, REF18, REF19, REF20, REF21,
 REF22, REF23, REF24, REF25, REF26, REF27, REF28, REF29, REF30, REF31,
 REF32, REF33, REF34, REF35, REF36, REF37, REF38, REF39, REF40, REF41,
 REF42, REF43, REF44, REF45, REF46, REF47, REF48, REF49, REF50, REF51,
 REF52, REF53, REF54, REF55, REF56, REF57, REF58, REF59, REF60, REF61,
 REF62, REF63, REF64, REF65, REF66, REF67, REF68, REF69, REF70, REF71,
 REF72, REF73, REF74, REF75, REF76, REF77, REF78, REF79, REF80, REF81,
 REF82, REF83, REF84, REF85, REF86, REF87, REF88, REF89, REF90, REF91,
 REF92, REF93, REF94, REF95, REF96, REF97, REF98, REF99, REF100, REF101,
 REF102, REF103, REF104, REF105, REF106, REF107, REF108, REF109, REF110,
 REF111, REF112, REF113, REF114, REF115, REF116, REF117, REF118, REF119,
 REF120, REF121, REF122, REF123, REF124, REF125, REF126, REF127, REF128,
 REF129, REF130, REF131, REF132, REF133, REF134, REF135, REF136, REF137,
 REF138, REF139, REF140, REF141, REF142, REF143, REF144, REF145, REF146,
 REF147, REF148, REF149, REF150, REF151, REF152, REF153, REF154, REF155,
 REF156, REF157, REF158, REF159, REF160, REF161, REF162, REF163, REF164,
 REF165, REF166, REF167, REF168, REF169, REF170, REF171, REF172, REF173,
 REF174, REF175, REF176, REF177, REF178, REF179, REF180, REF181, REF182,
 REF183, REF184, REF185, REF186, REF187, REF188, REF189, REF190, REF191,
 REF192, REF193, REF194, REF195, REF196, REF197, REF198, REF199, REF200,
 REF201, REF202, REF203, REF204, REF205, REF206, REF207, REF208, REF209,
 REF210, REF211, REF212, REF213, REF214, REF215, REF216, REF217, REF218,
 REF219, REF220, REF221, REF222, REF223, REF224, REF225, REF226, REF227,
 REF228, REF229, REF230, REF231, REF232, REF233, REF234, REF235, REF236,
 REF237, REF238, REF239, REF240, REF241, REF242, REF243, REF244, REF245,
 REF246, REF247, REF248, REF249, REF250, REF251, REF252, REF253, REF254,
 REF255, REF256, REF257, REF258, REF259, REF260, REF261, REF262, REF263,
 REF264, REF265, REF266, REF267, REF268, REF269, REF270, REF271, REF272,
 REF273, REF274, REF275, REF276, REF277, REF278, REF279, REF280, REF281,
 REF282, REF283, REF284, REF285, REF286, REF287, REF288, REF289, REF290,
 REF291, REF292, REF293, REF294, REF295, REF296, REF297, REF298, REF299,
 REF300, REF301, REF302, REF303, REF304, REF305, REF306, REF307, REF308,
 REF309, REF310, REF311, REF312, REF313, REF314, REF315, REF316, REF317,
 REF318, REF319, REF320, REF321, REF322, REF323, REF324, REF325, REF326,
 REF327, REF328, REF329, REF330, REF331, REF332, REF333, REF334, REF335,
55 * END COMDECK YAGSC

56 * BEGIN COMDECK YAGEQ

57 EQUIVALENCE(AASC(1),X,YPAR),(AASC(2),R,YPAR),(AASC(3),Y,MPAR),

58 (AASC(4),U),(AASC(5),V),(AASC(6),RO),(AASC(7),DEL8H,RCSQ,MP),(AASC

59 (AASC(15),SIE),(AASC(16),PHB,OKLM,RPK),(AASC(9)

60 (AASC(18),MP),(AASC(19),P,P,P),(AASC(11),P,P,P),(AASC(12),UTIL,

61 UL,MPK,PU),(AASC(13),MTLE,VL,PHY,PM),(AASC(14),AQ,CP,ROL),(AASC(17

62 (AASC(19),U),(AASC(20),VU),(AASC(21),VU),(AASC(22),GRROR),(AASC(23),

63 ,GRRORG),(AASC(24),GRRORZ),(AASC(25),DLRBO1,Y13K),(AASC(26),GV9

64 (AASC(27),DLRBO2,VG),(AASC(28),GRSV),(AASC(29),GRSP,YUVEC,

65 (AASC(30),CTEMP,X24K),(AASC(31),ANCV),(AASC(32),AVX8V,X13K),(AASC(34),

67 AVYSV,X24K)

68 REAL X,MP,MPAR,MTIL

69 * END COMDECK YAGEO

70 * END COMDECK YSTORE

71 * BEGIN COMDECK ASTORE

72 COMMON/ASC/AT(1,100),FT(100)

73 DIMENSION IX(1),IY(1),IX2(1),IY2(1),XCO(1),YCO(1),CON(1)

74 EQUIVALENCE(IX,IX1),(AT(2),IX2),(AT(3),IY1),(AT(4),IY2),(AT(5),XCO

75 ,X,FT,CON)

76 * END COMDECK ASTORE

77 * BEGIN COMDECK PCALL

78 COMMON/PCCALL/XCONVP,YCONVP,VUP,VLP

79 * END COMDECK PCALL

80 DIMENSION BCD(2),IYTITLE(1)

81 DIMENSION LABEL(26),IPLBL(26)

82 DATA ILABEL1,AH,1B,1C,1D,1E,1F,1G,1H,1I,1J,1K,1L,1M

83 1 1N,1O,1P,1Q,1R,1S,1T,1U,1V,1W,1X,1Y,1Z/

84 DATA IPLBL/17,18,19,20,21,22,23,24,25,33,34,35,36,37,38,39,40,41,

85 1 50,51,52,53,54,55,56,57/

86 DATA BCD/1H/

87 C FIND THE MINIMUM AND MAXIMUM (QMN AND QMX) OF THE QUANTITY

88 TO BE PLOTTED

89 C

90 QMN=1,E30

91 QMX=QMN

92 CALL START

93 DD 20 J=2,JP1

94 DD 10 I=1,ISAR

95 GMN=AMAX1(CD(IJ),QMN)

96 GMX=AMAX1(CD(IJ),QMX)

98 10 I=J+NO

99 CALL LOOP

100 20 CONTINUE

101 C SET THE CONTOUR INCREMENT, THE BASIC ALGORITHM IS BASED ON

102 THE DESIRE TO HAVE THE CONTOUR INCREMENT AN INTEGRAL POWER

104 OF TWO AND THE CONTOURS THEMSELVES INTEGRAL MULTIPLES OF
THIS POWER OF TWO, THESE FACTS INSURE THAT THE TWO PLOTS
VERY NEARLY THE SAME WILL HAVE EXACTLY THE SAME CONTOUR
VALUES, FIRST MAKE AN INITIAL GUESS FOR THE CONTOUR
INCREMENT (DQ) SO THAT THERE ARE AT LEAST TEN CONTOURS,

DQ*(QMX-QMN)/10,

IF THE INCREMENT IS ZERO OR NEGATIVE, BYPASS THE PLOT
IF(DQ.LE.0.) RETURN
FIND A POWER OF TWO JUST LESS THAN DQ BUT AT LEAST 2**33
WHICH IS ABOUT 10**10

TEST=2.*TEST
IF(TEST.GE.DQ) GO TO 40
TEST=2.*TEST
GO TO 30
CONTINUE

SET THE CONTOUR INCREMENT TO IT
DQ=.5*TEST

NOW DETERMINE THE CONTOUR VALUES, K WILL BE THE FINAL NO.
OF CONTOURS, THE FIRST CONTOUR IS THE NEAREST INTEGRAL
MULTIPLE OF DQ LESS THAN QMN.

50 CONTINUE
ITEST=QMN/DQ
IF(ITEST.LT.0.) ITEST=ITEST+1
CON(1)=FLOAT(ITEST)*DQ
DO 60 K=2,27
IF(CON(K-1).GT.QMX) GO TO 70
IF(K.LE.27) GO TO 60
CON(K)=CON(K-1)+DQ
60 CONTINUE

TOO MANY CONTOURS, GO INCREASE DQ AND TRY AGAIN.
GO TO 80
70 CONTINUE
K=K+1

IF THERE ARE MORE THAN 20 CONTOURS, INCREASE DQ AND GO CALCULATE
THE CONTOURS AGAIN.

IF(K.LE.20) GO TO 90
80 CONTINUE
DQ=2.*DQ
GO TO 50
90 CONTINUE

ADVANCE THE FILM TO THE NEXT FRAME
CALL ADV(1)
162 C
163 C SET THE HEIGHT OF THE FIRST LINE OF THE CONTOUR LABELS
164 C
165 C CALL LINCNT(6)
166 C
167 C LIST THE CONTOURS ON THE PLOT
168 C
169 C DO 100 KK=1,K
170 C WRITE(IFD,250) ILABEL(KK), CON(KK)
171 100 CONTINUE
172 C WRITE(IFD,260) QMN,QMX
173 C
174 C LABEL THE TYPE OF PLOT
175 C
176 C CALL LINCNT(58)
177 C WRITE(IFD,270) (ITITLE(I),I=1,NWT)
178 C WRITE(IFD,280) JNM,NAME, T,NCYC
179 C
180 C PUT CONTOUR INFORMATION OUT TO CYCLE SUMMARY
181 C
182 C IPDX=6
183 C WRITE(IPDX,280) (ITITLE(I),I=1,NWT)
184 C WRITE(IPDX,300) QMN,QMX, CON(1), CON(K=1), DO
185 C
186 C LOOP TO DO THE ACTUAL PLOT
187 C
188 C CALL START PULLS IN ROWS 1,2, AND 3 AND LEAVES IJ POINTING AT
189 C ROW 2
190 C
191 C CALL START
192 C DO 240 J=2,JBAR
193 C
194 C CALL LOOP AT THIS POINT SHIFTS THE ROWS IN SCM UP ONE SO THAT
195 C IJ POINTS AT ROW J+1 INSTEAD OF ROW J
196 C
197 C CALL LOOP
198 C DO 230 I=1,IM1
199 C IPJ=IJ+NO
200 C IPJM=IJM+NO
201 C
202 C NM0 SIGNIFIES THAT XCO AND YCO HAVE NOT BEEN COMPUTED FOR THIS
203 C I AND J
204 C
205 C NM0
206 C
207 C LOOP OVER ALL OF THE CONTOUR VALUES
208 C
209 C DO 220 KK=1,K
210 C
211 C SET FLAGS,
212 C
213 C CONSIDER THE FOUR CELLS (I,J),(I+1,J),(I,J+1), AND (I+1,J+1)
214 C AND LET THEM BE DENOTED CELLS 1,2,3, AND 4, RESPECTIVELY.
215 C THEN, KN IS ZERO IF THE QUANTITY TO BE PLOTTED IN CELL N
216 C IS GREATER THAN THE CURRENT CONTOUR VALUE, AND 1 IF
217 C THE QUANTITY TO BE PLOTTED IS LESS THAN OR EQUAL TO
218 C THE CURRENT CONTOUR VALUE,
NOTE THAT WHAT IS BEING CONSIDERED IS THE QUADRILATERAL FORMED BY
CONNECTING THE CENTERS OF THE FOUR CELLS IN THE ORDER
1=2=4=3=1, LET THIS BE KNOWN AS THE CURRENT CONTOUR AREA.

K1#0
K2#0
K3#0
K4#0

IF(CQ(IJM),LE,CON(KK)) K1#1
IF(CQ(IPJM),LE,CON(KK)) K2#1
IF(CQ(NIJ),LE,CON(KK)) K3#1
IF(CQ(IPJ),LE,CON(KK)) K4#1

IF ALL THE FLAGS ARE 1 OR 0, THEN THE CURRENT CONTOUR IS EITHER
GREATER THAN OR LESS THAN OR EQUAL TO THE PLOT QUANTITIES
AT ALL FOUR CORNERS OF THE CURRENT CONTOUR AREA, THUS, THE
CURRENT CONTOUR DOES NOT CROSS THE CURRENT CONTOUR AREA,
GO ON TO THE NEXT CONTOUR.

IF(K1*K2*K3*K4,NE,0,OR,K1+K2+K3+K4,EQ,0) GO TO 220

CALCULATE THE CENTERS OF CELLS 1,2,3, AND 4 (THE VERTICES OF THE
CURRENT CONTOUR AREA) IF THIS HAS NOT ALREADY BEEN DONE,

IF(N.GT,0) GO TO 130
IJB=IJM
IJ=IJM
DO 120 JJ=1,2
DO 110 II=1,2
IPJB=IJB+NQ
IPJIA=IJA+NQ
N=N+1

XCO(N),X*(X(IPJB)+X(IPJ)+X(IJA)+X(IJB))
YCO(N),Y*(Y(IPJB)+Y(IPJ)+Y(IJA)+Y(IJB))

IJA=IPJ
IJ=IJB
IJB=IJA
120 IJA=IJP
130 CONTINUE

DETERMINE WHICH SIDES OF THE CURRENT CONTOUR AREA THE CURRENT
CONTOUR CROSSES,
LL COUNTS HOW MANY SIDES THAT HAVE BEEN FOUND TO BE CROSSED
LL#0

SEE IF THE LEFT SIDE IS CROSSED

IF(K1*K3,NE,1) GO TO 140

YES, ICI AND IC2 ARE THE VERTEX NUMBERS OF THE CURRENT CONTOUR
AREA THAT BOUND THE SIDE CROSSED, IJ1 AND IJ2 ARE THE INDICES
OF THE MESH CELLS CONTAINING THESE VERTICES.

ICI=1
IC2=3
GO SET UP PLOT COORDINATES FOR THIS CROSSING

GO TO 170

SEE IF THE BOTTOM IS CROSSED BY THE CONTOUR

140 IF(K1+K2,NE,1) GO TO 150

YES

IC1=1
IC2=2
IJ1=IJM
IJ2=IPJM
GO TO 170

SEE IF RIGHT IS CROSSED

150 IF(K2+K4,NE,1) GO TO 160

IC1=2
IC2=4
IJ1=IPJM
IJ2=IPJ
GO TO 170

SEE IF TOP IS CROSSED

160 IF(K3+K4,NE,1) GO TO 220

IC1=3
IC2=4
IJ1=IJ
IJ2=IPJ

INCREMENT THE NO. OF SIDES CROSSED

170 LL=LL+1

CONTOUR LIES BETWEEN CO(IJ1) AND CO(IJ2). CALCULATE HOW FAR ALONG THE LINE CONNECTING VERTICES IC1 AND IC2 THE INTERSECTION SHOULD ACTUALLY LIE.

XX=(XCO(II)-CO(IJ1))/(CO(IJ2)-CO(IJ1))

IX1(II) AND IY1(II) ARE THE RASTER COORDINATES OF THE POINT OF INTERSECTION LL

FOR INTERSECTION LL

IF(IY1(II),GT,ITY,OR,IY1(II),LT,IYT) GO TO 210
IF(I1(LL).LT.IXL.OR.(I1(LL).GT.IXR)) GO TO 210

IF ONLY ONE INTERSECTION HAS BEEN LOCATED, GO BACK AND LOOK FOR ANOTHER

IF(LL.GE.2) GO TO 200

180 CONTINUE

IF IC1=2, RIGHT SIDE WAS INTERSECTED LAST, GO BACK AND GET THE TOP SIDE, IF IC1=3, TOP SIDE WAS INTERSECTED LAST, ALL SIDES HAVE BEEN CHECKED, GO ON TO NEXT CONTOUR.

GO TO (190,160,220), IC1

IC1=1, EITHER LEFT OR BOTTOM WAS LAST INTERSECTED, IF BOTTOM (IC2=2), GO BACK AND CHECK THE RIGHT, IF LEFT (IC2=3), GO BACK AND CHECK THE BOTTOM.

190 CONTINUE

IF(IC2.EQ.2) GO TO 150

GO TO 140

200 CONTINUE

TWO SIDES HAVE BEEN CROSSED, CONNECT THE POINTS OF INTERSECTION.

CALL DRV(I1(IY1,IY2)

PLOT THE LABEL ON THE FIRST INTERSECTION POINT

CALL PLT(I1(IY1),IPLBL(KK))

210 CONTINUE

START ON A NEW INTERSECTION PAIR IF BOTTOM WAS LAST SHIFT CHECKED, OTHERWISE, THERE ARE NOT TWO POSSIBLE INTERSECTIONS LEFT SO WE ARE DONE.

IF(IJ2.EQ.IPJM) GO TO 180

220 CONTINUE

IJM=IPJM

IJ=IPJ

230 IJP=IJP+NO

240 CONTINUE

DRAW THE PLOT FRAME AND LABEL IT

CALL TICBOX

RETURN

250 FORMAT(100XAL,2X1PE10,3)

260 FORMAT(98X3HMQN,2X1PE10,3/98X3HMQX,2XE10,3)

270 FORMAT(1H ,8A10)

280 FORMAT(1H ,16HCONTOUR PLOT OF ,6A10)

290 FORMAT(1H ,4AXA10,6A10,3X2HTL,1PE12,5,1X6HCYCLE=,15)

300 FORMAT(5H MIN=1PE12,5,H MAX=1PE12,5,3H L=E12,5,3H H=E12,5/1H ,16X

14H DQ=E12,5)
SUBROUTINE DEFINE

ROUTINE TO INITIALIZE PROBLEM INPUT VARIABLES

WRITTEN BY J. L. NORTON, LASL T-5, 1974

BEGIN COMDECK PARAM

COMMON/Pcom/Nscp1, ITABP, ITABXP, ITABYP, IFPB, NP1, NP2, NLCp1, NLCp2,
1 NLCp3, NLCp4, IFpms2

END COMDECK PARAM

BEGIN COMDECK YASC

LOGICAL RESTRT, FILM, PAPER, TURB

REAL LAM, MU

COMMON/Ysc1/Asc(9600)

COMMON/Ysc2/AA(1), ANC, AG, AGFAC, AMH, BO, COLAMU, CYL, DR, DT, DTC, DTFAC,
1 DTO(10), DTOC(18), DTOS, DTVP, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPYB, FIPYT, FIXL, FIXR, FYB, FYT, FREZXR, GR, GREL, GZ, GZIP, I, IBAR,
3 IDTO, IJ, IJH, IJP, IIMH, IPXL, IPXR, IPYB, IPYT, IP1, IP2, IISC2, IISC3, ITV,
4 IUNF, IXL, IXR, IYB, IYT, J, JBAR

COMMON/Ysc2/Jcen, JP1, JP2, JP4, JUNF, JUNFO2, KXI, LAM, LPS, MU, NAME(8),
1 NCV, NLC, NPS, NPT, NO, NOI, NO12, NSC, NUMIT, ZORIG, OM, OMCYL, PXCONV,
2 PXL, PXR, PYB, PYC, PYCV, PYT, RDT, REZRN, RESR, REZY, RIBAR, RIBJB,
3 FREZ, FREZB, ROMFR, T, THIRD, NCLST, TOUT, TWFN

COMMON/Ysc2/TUDT, TUSI, NCO, TNEG, TNESG, TUSV, TURB, TTOP, PRITE, PBTOM,
GR IS THE RADIAL COMPONENT OF THE BODY FORCE FELT BY THE ENTIRE PROBLEM
GR=0.

GZ IS THE AXIAL COMPONENT OF THE BODY FORCE FELT BY THE ENTIRE PROBLEM (USUALLY GRAVITY IN THE -Z DIRECTION)
GZ=9.80.

FREXR IS THE RATIO OF ZONE DR=S AS ONE MOVES AWAY FROM THE REGION OF CONSTANT ZONING TOWARD THE RIGHT BOUNDARY
FREXR=1.

FREYT IS THE RATIO OF ZONE DZ=S AS ONE MOVES AWAY FROM THE REGION OF CONSTANT ZONING TOWARD THE TOP BOUNDARY
FREYT=1.

FREZYT IS THE RATIO OF ZONE DZ=S AS ONE MOVES AWAY FROM THE REGION OF CONSTANT ZONING TOWARD THE BOTTOM BOUNDARY
FREZYT=1.

YB IS THE ACTUAL BOTTOM OF THE GRID
YB=0.

REZY0 IS THE CENTER OF THE REGION OF UNIFORM ZONING, EITHER YB OR REZY0 MAY BE SPECIFIED
REZY0=0.

REZRON IS THE INITIAL DENSITY OF THE AMBIENT ATMOSPHERE AT Z=REZY0
REZRON=0.01

REZSIE IS THE SPECIFIC INTERNAL ENERGY OF THE AMBIENT ATMOSPHERE
REZSIE=2.1E10

IBAR IS THE NUMBER OF REAL ZONES IN THE RADIAL DIRECTION
IBAR=0

JBAR IS THE NUMBER OF REAL ZONES IN THE AXIAL DIRECTION
JBAR=0

IUNF IS THE NUMBER OF UNIFORM ZONES IN THE RADIAL DIRECTION
IUNF=0

JUNF IS THE NUMBER OF UNIFORM ZONES IN THE AXIAL DIRECTION
JUNF=0
JCEN IS THE J VALUE ABOVE AND BELOW WHICH THERE ARE AN EQUAL NUMBER OF UNIFORM ZONES

JCEN=0

DR IS THE UNIFORM RADIAL MESH SPACING

DR=0.

DZ IS THE UNIFORM AXIAL MESH SPACING

DZ=0.

CYL IS THE GEOMETRY INDICATOR, CYL=1, FOR CYLINDRICAL GEOMETRY

CYL=0, FOR PLANE (SLAB) GEOMETRY

GRDVEL IS THE REZONE INDICATOR, GRDVEL=0, FOR EULERIAN

GRDVEL=1, FOR LAGRANGIAN

GRDVEL=2, FOR ALE

ANC IS THE NODE COUPLER CONSTANT

ANC=0.85

AOFAC IS A FACTOR USED IN CALCULATING THE COURANT TIMESTEP

AOFAC=2

AOFAC, ABM, AND BB ARE REZONE COEFFICIENTS WHICH DETERMINE THE TYPE OF DIFFERENCING USED IN CALCULATING THE FLUXING TERMS. AOFAPPLIES TO THE MOMENTUM EQN, ABM TO THE MASS AND ENERGY EQNS, AND BB TO ALL THREE. SOME EXAMPLES OF THEIR SIGNIFICANCE ARE

<table>
<thead>
<tr>
<th>AOFAC</th>
<th>ABM</th>
<th>BB</th>
<th>DIFFERENCING</th>
<th>SIGNIFICANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>CENTERED</td>
<td>CENTERED</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>FULL DONOR</td>
<td>FULL DONOR</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>INTERPOLATED</td>
<td>INTERPOLATED</td>
</tr>
</tbody>
</table>

NOTE THAT THE EQNS ARE UNSTABLE IN THE FIRST CASE.

AGM=1

ABM=1

BB=0

KXI IS A PARAMETER GOVERNING THE TREATMENT OF VISCOSITY.

KXI=1 ALLOWS MU AND LAM TO BE TREATED AS NUMERICAL VISCOSITY COEFFICIENTS

KXI=0 ALLOWS MU AND LAM TO BE TREATED AS TRUE PHYSICAL VISCOSITY COEFFICIENTS

KXI=1 CAUSES THE CODE TO CALCULATE ITS OWN NUMERICAL VISCOSITY COEFFICIENTS BASED ONLY UPON THE RESTRICTION THAT THE RATIO OF THE MU AND LAM COMPUTED INTERNALLY BY THE CODE IS EQUAL TO THE RATIO OF THE MU AND LAM READ IN
GZIP IS THE BODY FORCE FELT BY THE PARTICLES, IT IS NOT NECESSARILY
EQUAL TO GZ.

GZIP=980.

ZORIG IS THE NO. OF FIREBALL RADI TO HOLD THE OUTER BOUNDARY
AWAY FROM THE FIREBALL

ZORIG=6.

T IS THE PROBLEM START TIME

T=0.

DT IS THE INITIAL Timestep

DT=.001

NCLST IS THE CYCLE NO. AT WHICH TO HALT THE RUN

NCLST=99999

TWFIN IS THE REAL TIME AT WHICH TO TERMINATE THE RUN

TWFIN=1.030

DTO AND DTOC ARE OUTPUT FREQUENCY CONTROL ARRAYS, IF T.LE.DT0C(I),
EDITS AND PLOTS OCCUR EVERY DTO(I) PROBLEM SECONDS, IF
DTOC(I)=1,LT.T.LE.DT0C(I),THEN OUTPUT OCCURS EVERY DTO(I)
PROBLEM SECONDS, A MAXIMUM OF 10 DTO,DTOC PAIRS MAY BE
SPECIFIED.

DO(1)=1,830

DO(I)=81,E30

DO 10 II=2,810

10 DTOC(II)=0.

NCQ IS THE CYCLE ON WHICH TURBULENCE IS
LESS THAN 8, THERE IS NO TURBULENCE,

NCQ=1

IF TURB IS ,FALSE,, THE TURBULENCE IS CURRENTLY OFF

TURB*,FALSE.

QLEVEL, TUGI, TUSI, AND TO ARE ALL TURBULENCE QUANTITIES

QLEVEL=02

TUGI=0.

TUSI=0.

TQ=0.

A PARTICLE CAN BE MOVED NO MORE THAN
SQRT(2)*WMA*SQRT(2*M*SIGMA*DT) DUE TO THE TURBULENT
DIFFUSION EFFECT DURING ANY ONE CYCLE

WMA*SQRT(2).

ANY PARTICLE WITH XJ,LE,RHINEF WILL NOT BE SUBJECT
TO TURBULENT DIFFUSION

RHINEF=50.

TSTRTD IS THE TIME AT WHICH TO START TURBULENT DIFFUSION

TSTRTD=1.

IST IS THE NUMBER OF PARTICLES TO FOLLOW AS A FUNCTION OF TIME,
IF IST,LT,0, NO PARTICLES WILL BE FOLLOWED.

IST=1

JDNMP IS THE FREQUENCY OF DUMP CYCLES

JDNMP=999999

RETURN

END

-----------------------------------------------------------------------------------

SUBROUTINE DMP(FWA,LWA,IFILE)

ROUTINE TO DUMP SCM FROM FWA TO LWA WITH THE OUTPUT DIRECTED
TO LOGICAL UNIT IFILE

WRITTEN BY J.L.NORTON, LASL T=3, 1975

INTEGER FWA,GETIT
DIMENSION IDUMP(4)

CHECK FOR ERRORS

IF(FWA,LE,LWA) GO TO 10

YES, PRINT MESSAGE AND QUIT.

WRITE(IFILE,100) FWA,LWA
RETURN

CONTINUE

ALL OK, PRINT DUMP HEADER.

WRITE(IFILE,110) FWA,LWA

IF FWA IS THE ADDRESS OF THE FIRST WORD TO BE PRINTED ON THE
CURRENT LINE
ILW is the address of the word currently being processed

IFW=FMA

ILW=FMA

ICLW is the contents of the word whose address is ILW

ICLW=GETIT(ILW)

IDUMP contains the words to be printed on the current line (a maximum of four).

ISUB is the no. of the last location of IDUMP that was filled.

ISUB=1

IDUMP(I)=ICLW

20 CONTINUE

EXAMINE THE NEXT WORD

ILW=ILW+1

SEE IF WE ARE DONE

IF(ILW,LE,LWA) GO TO 30

YES, FLUSH IDUMP IF NECESSARY AND QUIT.

IF(ISUB,LE,0) RETURN

WRITE(IFILE,120) IFW,(IDUMP(I),I=1,ISUB)

RETURN

30 CONTINUE

NO, SAVE THE CONTENTS OF THE NEXT-TO-THE-LAST WORD AND

GO GET THE CONTENTS OF THE CURRENT WORD.

ICNLW=ICLW

ICLW=GETIT(ILW)

SEE IF THE LAST WORD AND THE NEXT-TO-THE-LAST WORD ARE THE SAME

IF(ICLW,NE,ICNLW) GO TO 80

YES, WE WILL GO INTO REPEITION MODE. FIRST FLUSH IDUMP

IF NECESSARY.

ISUB=ISUB+1

IF(ISUB,LE,0) GO TO 40

WRITE(IFILE,120) IFW,(IDUMP(I),I=1,ISUB)

ISUB=0

40 CONTINUE

IFW is now the address of the first word of the repetition group

IFW=ILW=1

SCAN FORWARD UNTIL THE END OF THE REPEITION GROUP IS LOCATED
IF WP = IF WP + 2
DO 50 I=IF WP, LWA
I CL W = GET IT (I)
IF (I CL W .NE. IC NL W) GO TO 60
50 CONTINUE

C ALL THE REST OF THE REQUESTED DUMP REGION LIES WITHIN THE
C REPETITION GROUP
I CL W = LWA
GO TO 70
60 CONTINUE

C I LW IS THE ADDRESS OF THE LAST WORD IN THE REPETITION GROUP
I LW = I + 1
70 CONTINUE

C WRITE OUT THE REPETITION GROUP
C WRITE (IFILE, 130) IF W, I LW, IC NL W
C GO BACK AND CONTINUE PROCESSING UNLESS WE ARE DONE
C I LW = I LW + 1
C IF (I LW .GT. LWA) RETURN
C I LW = ILW
C ISUB = 1
C IDUMP (I) = ICL W
GO TO 20
80 CONTINUE

C LAST AND NEXT-TO-THE-LAST WORD ARE DIFFERENT. SEE IF THE IDUMP
C BUFFER IS FULL.
C IF (ISUB .LT. 4) GO TO 90
C YES, FLUSH IT,
C WRITE (IFILE, 120) IF W, (IDUMP (I), I = 1, 4)
C IF WM I LW
C ISUB = 0
C 90 CONTINUE
C ADD THE CURRENT WORD TO THE IDUMP BUFFER AND CONTINUE
C ISUB = ISUB + 1
C IDUMP (ISUB) = ICL W
GO TO 20

100 FORMAT (1 H, $00ERROR IN DMP ARGUMENTS, DMP BYPASSED, FWA AND LWA
110 14HARE $210)
110 FORMAT (1 H, 17H DUMP OF SCM FROM $06,4 H TO $06,7 /
117 120 FORMAT (1 H, $06,5X4025)
118 130 FORMAT (1 H, $06,6 H THRU $06,13 H ALL CONTAIN $020)
139 END
SUBROUTINE DMPK(N, PACK)

ROUTINE TO DUMP EXCHANGE PACKAGE

WRITTEN BY LARRY RUDINSKI AND JERRY MELENDEZ, LASL C=4

MODIFIED TO CORRECTLY PICK UP C(A0) = C(A7) BY J. L. NORTON, LASL TO3

DIMENSION PACK(1), PARCEL(4)

DIMENSION ISAVE(8)

DIMENSION NPC(8), NAC(8), NBC(8), NXC(8)

DIMENSION IAA(9), IA1(8), IA3(2)

INTEGER PACK, PARCEL

INTEGER PACK1

INTEGER GETIT, SHIFT, AND

DATA NPC/

DATA NAC/

DATA NBC/

DATA NXC/

DATA IAA/ SHC(A0), SHC(A1), SHC(A2), SHC(A3), SHC(A4), SHC(A5), SHC(A6),

1 SHA(A7)/

DATA IA3/7H OUT OF 6H RANGE/

IF(N.EQ.0) N=6

PRINT CAUSE OF ABORT

IARG=SHIFT(PACK(4), =36)

CALL MODE(IARG, N)

GET THE FIELD LENGTH

IAA(9)=SHIFT(PACK(3), =36)

IF FIELD LENGTH IS GARBAGED, SET TO MAXIMUM

IF(IAA(9), GT, 150077B) IAA(9)=150077B

PICK UP AND PRINT REGISTERS, ETC.

DO 30 I=1, 8

PICK UP B REGISTER I=1

NB=AND(PACK(I), 777777B)

PICK UP A REGISTER I=1

NA=AND(SHIFT(PACK(I), =18), 777777B)

GET C(AN)

IF(NA, GE, IAA(9)) GO TO 10

ISAVE(I)=GETIT(NA)

10 CONTINUE
SUBROUTINE DVMM(VMAX, VMH, VMH, DRMIN, DZMIN, DMAX, DZMAX)

ROUTINE TO CALCULATE MAXIMUM VELOCITY AND MAXIMUM AND MINIMUM ZONE SIZES
BEGIN COMDECK PARAM

COMMNP COM/NSCP1, ITABP, ITABXP, ITABY, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, NLMZ

END COMDECK PARAM

BEGIN COMDECK YSTORE

BEGIN COMDECK YADIM

DIMENSION X(1), XPAR(1), XPAR(1), YPAR(1), YPAR(1), U(1), UG(1), DELSM(1),
1, V(1), VG(1), RO(1), SIE(1), MP(1), RMP(1), RSG(1), C(1), ETIL(1), RVL(1),
2, UP(1), UP(1), UP(1), UP(1), UP(1), UP(1), UP(1), UP(1), UP(1), UP(1),
3, VL(1), ROL(1), AVSV(1), AVSV(1), TLSRO(1), TLSRO(1), TLSRO(1), TLSRO(1),
4, SIG(1), TSU(1), GROR(1), GRROZ(1), GROR(1), GROR(1), MTIL(1),
5, CONC(1), CTMP(1), ANCU(1), ANCV(1), GR8V(1), GS8V(1), X13K(1), X24K(1),
6, Y13K(1), Y24K(1), X13K(1), X24K(1), DLKSM(1), ARA(1)

END COMDECK YADIM

BEGIN COMDECK YAQSC

LOGICAL RESTRT, FILM, PAPER, TURB

REAL LAM, MU

COMMON/Y8C1/AASC(9600)

COMMON/Y8C2/AA(1), ANC, AO, A0PAC, A0M, BM, COLAMU, CYL, DR, DT, DTC, DTFC,
1, DT0(18), DT0C(18), DT0E, DT0P, DTV, DZ, EM10, EPS, FIPX, FIPX,
2, FIPY, FIPYT, FIPX, FIPX, FITY, FITY, FREZXR, GR, GRVEL, DZ, GEP, I, IBAR,
3, IDG, IJ, JX, JY, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ, JZ,
4, IUNF, IXL, IYR, IYT, J, JBAR


COMMON/Y8C3/NC, NPS, NPT, NC, NQ, NQ, NQ, NQ, NQ, NQ, NUMIT, ZORIG, OM, OM, CMYL, PXCONV

COMMON/Y8C31/IXL, IXR, IYT, J, JBAR

COMMON/Y8C311/JP1, JP2, JP4, JUNF, JUNFO, KXI, LAM, LPM, BU, NAME(16)

COMMON/Y8C3111/IXL, IXR, IYT, J, JBAR

CQUIVALENCE(AASC(1), X, XPAR), (AASC(2), R, YPAR), (AASC(3), Y, MPAR),
1, AAC(4), U, AAC(5), V, AAC(6), RO, AAC(7), DELSM, RSGQ, MP, (AASC(8),
2, E, ETIL, AREA, XR13K),
3, AAC(15), BIE, (AASC(15)), PM8, DDL8M, RPM, (AASC(19),
4, J, RVL0), (AASC(18)), ANM, V0, (AASC(18)), PL0, EP0, UP0, (AASC(12), UTIL0,
5, U, PMX, NPU, (AASC(13), V0, V0, NPU, V0, (AASC(14)), QC0, ROL0, (AASC(17),
6, CAPM, UG, (AASC(18), UQ0), (AASC(19), SIG), (AASC(20), TUS), (AASC(21),
7, GROR, (AASC(22), GROR0), (AASC(23), TLSRO, Y13K), (AASC(24), GS8V
8, (AASC(25), TLSRO, VG), (AASC(26), GS8V), (AASC(27), GROR, DDL8V,
9, Y24K), (AASC(28), MTIL0, (AASC(29), CONC), (AASC(30), CTMP, X24K),
0, (AASC(31), ANCU), (AASC(32), ANCV), (AASC(33), AVSV, X13K), (AASC(34),
1, AVSV, X24K)
REAL M, MP, MPAR, MTIL

* END COMDECK YAGED *----

* END COMDECK YSTORE *----

C DR(Z)MIN(MAX) IS THE MINIMUM (MAXIMUM) OF THE MAGNITUOE OF
CELL SIDES IN THE R (Z) COORDINATE DIRECTION

C

DRMIN=1.E+20
DZMIN=1.E+20
DRMAX=0.
DZMAX=0.

C VMAX IS THE MAXIMUM GRID VELOCITY MAGNITUOE ALONG EITHER
COORDINATE AXIS

C

VMAX=0.
IVH=0.
JVM=0.

CALL START

C LOOP OVER ALL REAL ZONES
C

DO 30 J=2,JP1
DO 20 I=1,IBAR
IPJ=IJ+NG
IPJP=IPJ+NG
VMAX=MAX(VMAX,ABS(U(IJ)),ABS(V(IJ)))
IF(VMAX.EQ.VMAX) GO TO 10
IVMI
JVM=J
VMAX=VMAX

10 CONTINUE

C DETERMINE THE FOUR VERTICES OF CELL (I,J)
C

(X1,Y1) IS VERTEX (I+1,J) (VERTEX 1)
(X2,Y2) IS VERTEX (I+1,J+1) (VERTEX 2)
(X3,Y3) IS VERTEX (I,J+1) (VERTEX 3)
(X4,Y4) IS VERTEX (I,J) (VERTEX 4)

C

XI=X(IPJ)
X2=X(IPJP)
X3=X(IPJ)
X4=X(IPJ)
Y1=Y(IPJ)
Y2=Y(IPJP)
Y3=Y(IPJP)
Y4=Y(IPJP)

C DETERMINE THE SQUARE OF THE LENGTH OF EACH SIDE
C

(X,Y)NM IS THE SQUARE OF THE LENGTH OF THE SIDE BOUNDED BY
VERTICES N AND M

C

X14=(X1-X4)**2+(Y1-Y4)**2
X23=(X2-X3)**2+(Y2-Y3)**2
Y21=(X2-X1)**2+(Y2-Y1)**2
Y34=(X3-X4)**2+(Y3-Y4)**2
FUNCTION ERF(Z)

ROUTINE TO CALCULATE THE STANDARD ERROR FUNCTION

STANDARD LIBRARY SUBPROGRAM = LOS ALAMOS SCIENTIFIC LABORATORY
DOCUMENTED BY J. L. NORTON, LA-8L TD-3, 1972

DIMENSION P(7,2), Q(6,2)
DIMENSION A(14), B(12)
EQUIVALENCE (A,P), (B,Q)
X=ABB(Z)

IF ARGUMENT IS ZERO, ERF IS ZERO

ERF=0.
IF(X,EQ.0.) RETURN

DATA (A(I), I=1,14)/1,1283791670955, 3419750591854,
1 ,629060145520E1, 1 ,12382023274723E1, 1 ,11986242418382E2,
2 ,7653732687825E4, 2 ,2535682058342E5, 2 ,99999707683738,
3 ,475179432895, 3 ,057349601594, 3 ,4407839213875,
4 ,1066419750781, 4 ,12636031836273E1, 4 ,1149393366616E6/

DATA (B(I), I=1,12)/1,36359916427762, 1,52205830591727E1,
1 ,3061303568519F2, 1,46856639020338E4, 1,15601995561434E4,
2 ,62143554992876E6, 2,6015349994799, 2,992955675358,
3 ,19684584582864, 3 ,7925079527606, 1 ,1693702005137,
4 ,23968882855053E1/

IF ARGUMENT IS GREATER THAN 5.5, ERF IS UNITY

10 ERF=SIGN(1., Z)

USE RATIONAL APPROXIMATION TO COMPUTE ERF

IF(X,GE,5.5) RETURN
J=1.
IF(X, GT, 1.5) J=2
TWO SEPARATE RATIONAL APPROXIMATIONS ARE USED, BOTH HAVE BEEN
DERIVED USING A PROGRAM BASED ON MAHERLY'S SECOND DIRECT
METHOD DESCRIBED IN JOURNAL OF THE ACM, VOL. 10, NO. 3

\[
ERF(D*EXP(-X*X)\times(P(1,J)+U\times(P(2,J)+U\times(P(3,J)+U\times(P(4,J)+U\times(P(5,J)+U\times(P(6,J)+U\times(P(7,J))))))))/1\times(U\times(Q(1,J)+U\times(Q(2,J)+U\times(Q(3,J)+U\times(Q(4,J)+U\times(Q(5,J)+U\times(Q(6,J))))))))*SIGN(1,J)
\]
RETURN
END

SUBROUTINE PILMCO

ROUTINE TO UPDATE GRID LIMITS AND PARTICLE QUANTITIES
ORIGINALLY WRITTEN BY A. A. AMBREN, LABL 1973
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL 1973

--- BEGIN COMMON PARAM ---
COMMON/PCON/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFMLC

--- END COMMON PARAM ---

--- BEGIN COMMON Y_STORAGE ---

DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UC(1), DLSM(1)
1 IP1, VG(1), RO(1), SIE(1), MP(1), RPM(1), RENSQ(1), EN(1), ETIL(1), RVOL
2 K(1), RM(1), VPC(1), P(1), PL(1), UP(1), UTIL(1), UL(1), CQ(1), VTIL(1)
3 VL(1), ROL(1), AVXY(1), AVYVS(1), DLSRO(1), SLRHO(1), CAPGAM(1), TUG
4 SIG(1), TUS(1), GROR(1), GROG(1), GROD(1), TUGC(1), MTIL(1)
5 CONC(1), CTEMP(1), ANCU(1), ANCV(1), GRSV(1), GZSV(1), X13K(1), X24K(1)
6 Y13K(1), Y24K(1), XR13K(1), XR24K(1), DKL(1), AREA(1)

--- END COMMON PARAM ---

--- BEGIN COMMON YAOSC ---
LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAH, MU

COMMON/YSCI/AAASC(NSCP1)
COMMON/YSCI/AAASC(9600)
COMMON/YSCI2/AAASC(96000)

--- END COMMON YAOSC ---
38 2 \text{VV}, \text{XCONV}, \text{XL}, \text{XR}, \text{YB}, \text{YCONV}, \text{YT}, \text{PTPOLD}, \text{DTSV}, \text{DTLAG}, \text{FIYB}, \text{IYBO}, \text{YCNVLD},
39 \text{IN}, \text{FIXRO}, \text{FIXLO}, \text{IXRO}, \text{IXLO}, \text{JSVM}, \text{GMY}, \text{GMY}, \text{WMAX}, \text{JMY}, \text{T2}, \text{TLM},
40 \text{ROMFX}, \text{ROMFY}, \text{ROMFYB}, \text{JUMP}, \text{TWTHRD}, \text{TE}, \text{DTR}, \text{TMASS}, \text{DTSAV}, \text{DTCSAV}, \text{IDTV}
41 \text{DT}, \text{IDTV}, \text{IDTC}, \text{IDTC}, \text{CIRC}, \text{CIRC}, \text{TIS}, \text{POE}, \text{UMOM}, \text{VMOM}, \text{TMAX}, \text{TGMX}, \text{ITM}, \text{ITM}, \text{ITG}, \text{ITG}
42 6 \text{TMASS}, \text{WMAXEF}, \text{RMINEF}, \text{TSTRTD}
43 \text{COMMON/YSC2/ZZ}
44 \text{COMMON/YSC4/ITAB(ITABP)}
45 \text{COMMON/YSC4/ITAB(100R)}
46 \text{COMMON/YSC5/RESTRT,FILM,PAPER,IPD,IFD}
47 \text{**** END COMDECK YAGBC ****}
48 \text{BEGIN COMDECK YAGED ****}
49 \text{EQUIVALENCE(AASC(1),X,XPAR),(AASC(2),Y,MPAR),(AASC(3),Z,MPP),(}
50 \text{AASC(4),U),(AASC(5),V),(AASC(6),W),(AASC(7),DELSM,RCGO,MP),(AASC}
51 \text{X1,ETIL,AREA,XR13K),}
52 \text{Z2,(AASC(15),SIG),(AASC(16),PMBO,DKLBM,RMP),(AASC(9}
53 \text{Z3),ATVOL),(AASC(10),NR,VR),(AASC(11),P,P,E,U),(AASC(12),UTIL},
54 \text{Z4,UL,PX,PX),(AASC(13),VTIL,VL,PMV,PMV),(AASC(14),Q,CO,ROL),(AASC(17}
55 \text{Z5),(CAPGAM,UG),(AASC(18),TUG),(AASC(19),SIG),(AASC(20),TUB1),
56 \text{AASC(21),GRORD),(AASC(22),GRROD),(AASC(23),DLSRO1,Y13K),(AASC(24),GZSV}
57 \text{Z7),(AASC(25),DLSRO2,VG),(AASC(26),GZSV),(AASC(27),GRROP,TUVEC},
58 \text{Z8,Y24K),(AASC(28),MTIL),(AASC(29),CONC),(AASC(30),CTEMP,XR24K),}
59 \text{Z9,AASC(31),ANCU),(AASC(32),ANCY),(AASC(33),AVXSV,X13K),(AASC(34},
60 \text{Z0),(AASC(35),AVXSV,X13K)}
61 \text{REAL M,MP,MPAR,MTIL}
62 \text{**** END COMDECK YAGED ****}
63 \text{**** END COMDECK YSTORE ****}
64 \text{C FIND THE GRID LIMITS, MAX(X)=XX, MAX(Y)=YT, MIN(X)=XL=0, MIN(Y)=YB,}
65 \text{C}
66 \text{XL=0,0}
67 \text{YB=1,E+20}
68 \text{XR=YT=YT}
69 \text{CALL START}
70 \text{DO 20 J=2,JP2}
71 \text{DO 19 I=1,IP1}
72 \text{XR=AMAX1(XR,X(IJ))}
73 \text{YB=AMIN1(YB,Y(IJ))}
74 \text{YT=AMAX1(YT,Y(IJ))}
75 \text{I=J}
76 \text{CALL LOOP}
77 \text{20 CONTINUE}
78 \text{C VV IS USED IN SCALING VELOCITY VECTOR PLOTS,}
79 \text{VV = 9*MAX(X)/IBAR}
80 \text{C}
81 \text{VV=0.9*XR/IBAR}
82 \text{C}
83 \text{FIYB IS THE LOCATION OF MIN(Y) IN RASTER COUNTS}
84 \text{C}
85 \text{FIYB=916.0}
86 \text{C}
87 \text{XD IS THE RATIO OF GRID WIDTH TO HEIGHT}
88 \text{C}
89 \text{XD=XR/(YT=YT)}
90 \text{C}
91 \text{ONE WISHES TO MAKE THE PLOTS FILL THE FILM FRAME AS NEARLY AS}
92 \text{POSSIBLE; AT MOST THE PLOT CAN BE 1022 RASTER POINTS WIDE AND}
93 \text{900 RASTER POINTS TALL (LEAVING ROOM FOR LABELS AT THE BOTTOM}
94 \text{C
107}
AND A 16 POINT MARGIN AT THE TOP, AN IDEAL GRID, THEN, WOULD
HAVE XD = 1022/900 = 1.13556. IF XD > 1.13556, THE GRID IS DEFINED
AS WIDER THAN HIGH. IF XD < 1.13556, THE GRID IS DEFINED AS
HIGHER THAN WIDE. IN THE FORMER CASE, THE X COORDINATE RASTER
BOUNDS ARE SET TO (0, 1022) AND THE Y COORDINATE RASTER BOUNDS
TO (FIYT, 916) WHERE FIYT IS DETERMINED SUCH THAT THE X AND Y
SCALES ARE THE SAME. IN THE LATTER CASE, THE Y COORDINATE
RASTER BOUNDS ARE SET TO (16, 916) AND THE X COORDINATE RASTER
BOUNDS ARE DETERMINED SUCH THAT THE X AND Y SCALES ARE EQUAL
AND THE LEFT AND RIGHT MARGINS ARE THE SAME.

IF(XD < 1.13556) GO TO 30

GRID WIDER THAN HIGH, THE X RASTER BOUNDS ARE (FIXL, FIXR), THE
Y RASTER BOUNDS ARE (FIYT, FIYB).

FIXL = 0,
FIXR = 1022,
FIYT = 16, 1022/XD
GO TO 40

GRID HIGHER THAN WIDE

30 CONTINUE

FIXL = MAX(0, FIYT - 16)
FIXR = FIXL + 1022
FIYT = 16,
GO TO 40

40 CONTINUE

XCONV AND YCONV ARE FACTORS TO CONVERT FROM X AND Y CARTESIAN
COORDINATES TO RASTER COORDINATES

XCONV = (FIXR - FIXL) / (XR - XL)
YCONV = (FIYT - FIYB) / (YT - YB)

PROVIDE FIXED POINT VALUES OF THE RASTER BOUNDS

IF THERE ARE NO PARTICLES, WE ARE DONE

IF(NPT, EQ, 0) RETURN

THERE ARE PARTICLES, CALCULATE THE CARTESIAN PARTICLE BOUNDS
FROM THE GRID BOUNDS.

PXL = 0, 0
PYB = YB
PRX = XR
PYT = YT
FIPX = 916, 0
FIPY = FIYT
FIPX = FIXL
FIPY = FIXR
FIPY = FIXT
CONVERSION FACTORS FROM PARTICLE COORDINATES TO RASTER COORDINATES

\[
\begin{align*}
\text{PXCONV} &= (\text{FIPXR} - \text{FIPXL}) / (\text{PXR} - \text{PXL}) \\
\text{PYCONV} &= (\text{FIPYT} - \text{FIPYB}) / (\text{PYT} - \text{PYB})
\end{align*}
\]

PROVIDE FIXED POINT RASTER BOUNDS

\[
\begin{align*}
\text{IPXL} &= \text{FIPXL} \\
\text{IPXR} &= \text{FIPXR} \\
\text{IPYB} &= \text{FIPYB} \\
\text{IPYT} &= \text{FIPYT}
\end{align*}
\]

RETURN

---

SUBROUTINE GETOMG

ROUTINE TO CALCULATE VORTICITY

ORIGINALLY WRITTEN BY A.A. AMSEN, LASL T-3

MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-3, 1974

COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, ITFLMSZ

BEGIN CODE BLOCK PARAM

COMMON/PAOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, ITFLMSZ

BEGIN CODE BLOCK YSTORE

BEGIN CODE BLOCK YAGDIN

DIMENSION X(1), XPAR(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM(1),
1 V(1), VG(1), ROL(1), SIE(1), MP(1), RMP(1), RCSQ(1), E(1), ETIL(1), RVOL
2 (1), H(1), RM(1), VM(1), P(1), PL(1), UP(1), UTIL(1), UL(1), CG(1), VTI1(1),
3 VL(1), ROL(1), AVXS(1), AVYS(1), DLRSQ(1), DLRSQ(1), CAPPAG(1), TUQ
4 SIG(1), TM(1), GRM(1), GRRO(1), GRP(1), GRP(1), TVBQUE(1), MTILE(1),
5 CONC(1), TTEMP(1), ANCO(1), ANCV(1), GRVY(1), GZSV(1), X13K(1), X24K(1),
6 Y13K(1), Y24K(1), XR13K(1), XR24K(1), DKLSM(1), AREA(1)

END CODE BLOCK YAGDIN

BEGIN CODE BLOCK YAGSC

LOGICAL RESY, FILM, PAPER, TURB

REAL LAM, MU

COMMON/YSC1/ASC(NSCP1)

COMMON/YSC2/ASC(9600)

COMMON/YSC2/ASC(AA1A), ANC, A4, AS0FA, A0M, A0B, COLAMU, CYL, DR, DT, DTC, DTFAC,
1 DCO(10), DTC(10), DTO2, DTOB, DTOB, DTOB, DTOB, DTOB, DTOB, DTOB, DTOB, DTOB,
2 FIPB, FIPY, FIPX, FIPY, FIPX, FIPY, FIPX, FIPY, FIPX, FIPY, FIPX,
3 SIG, TM, IPX, IPY, IPX, IPY, IPX, IPY, IPX, IPY, IPX,
4 IUNF, IXL, IYB, INT, JBAR

COMMON/YSC2/JCEN, JP1, JP2, J4P, JUNF, JUNFO2, KXI, LAM, LPB, MU, NAME(8),
1 NCYC, NLC, NPS, NPT, NG, NQI, NQI2, NSEC, NUMIT, ZORIG, OMCY, OMCY, OMCY,
2 PXL, PXR, PYB, PYCNO, PPT, REZON, REZIE, REZIE, REZIE, REZIE, REZIE,
3 FREZY, FREZV, ROMFR, T, THIR, NCLST, TOUT, TWFIN

COMMON/YSC2/TQDI, TUSI, NCQ, TNG, TNGS, TUSV, TURB, TPOT, PRITE, PBOB, T

109
COMMON/YSC2/2Z
COMMON/YSC4/ITAB(ITABP)
COMMON/YSC5/RESTRT,FILM,PAPER,IPD,IFD

* ===== BEGIN COMDECK YAGEQ =====

EQUIVALENCE(AASC(1),X, XPAR),(AASC(2),R,Y PAR),(AASC(3),Y,MPAR),
1 AASC(4),U),(AABC(5),V),(AASC(6),RD),(AASC(7),DELSM,RC8Q,MP),(AASC
1 (8),E,ETIL,AREA,XR13K),
2 (AASC(15),STE),(AASC(16),PM8,DKLSM,RMP),(AASC(9
3 ),RVOl),(AASC(10),H,RM,VP),(AASC(11),P,PI,EP,UP),(AASC(12),UTIL,
4 UL,PMX,PU),(AASC(13),VTIL,VL,PMY,PV),(AASC(14),Q,KQ,ROL),(AASC(17
5 ),CAPGAN,UG),(AASC(18),TUQ),(AASC(19),SIG),(AASC(20),TUB),(AASC(
6 21),GRMOR),(AASC(22),GRMOR),(AASC(23),DBRO1,Y13K),(AASC(24),G125V
7 )),(AASC(25),DLSROG,VG),(AASC(26),GR5V),(AASC(27),GRROP,TUQVEC,
8 Y24K),(AASC(28),HTIL),(AASC(29),CONC),(AASC(30),CTEMP,M9124K),
9 AASC(31),ANCU),(AASC(32),ANCV),(AASC(33),AVX8V,UX8K),(AASC(34),
10 AVYSV,X24K)

REAL MP,MPAR,MTIL

* ===== END COMDECK YAGEQ =====

* ===== END COMDECK YSTORE =====

10 CALL START

WMAX=MEM10
GMM=1.E0
QM8=QM8

DO 30 J=2,JP1
DO 20 I=1,IBAR
IPJ=I+NO
IFJ,EQ,2) CO(IJM)=0.
IF(J,EQ,JP1) CO(IJP)=0.
IF(J,EQ,IBAR) CO(IJP)=0.
X1=X(IPJ)
Y1=Y(IPJ)
U1=U(IPJ)
V1=V(IPJ)
X2=X(IPJP)
Y2=Y(IPJP)
U2=U(IPJP)
V2=V(IPJP)
X3=X(IPJ)
Y3=Y(IPJ)
U3=U(IPJ)
V3=V(IPJ)
X4=X(IPJ)
Y4=Y(IPJ)
U4=U(IPJ)
V4=V(IPJ)

R1=1.25*RVOL(IJ)*R(IPJ)+R(IPJ)*R(IJP)+R(IJ)
C0(IJ)=R1*(U1+U4)*(X1+X4)+(V1+V4)*(Y1+Y4)+(U2+U1)*(X2+X1)+(V2+V1)
1*(Y2+Y1)+(U3+U2)*(X3+X2)+(V3+V2)*(Y3+Y2)+(U4+U3)*(X4+X3)+(V4+V3)*
SUBROUTINE MODE(IARG,NARG)
C
ROUTINE TO ANALYZE THE PROGRAM STATUS DESIGNATION
C
WRITTEN BY J.L. NORTON, LASL TO=3, 1973
C
DIMENSION ITROUB(12),ICAUSE(2,12)
C
LOGICAL ITROUB
INTEGER I,J
DATA ICAUSF/2HUNDERFLOW, 1H, 8HOVERFLOW, 1H, 18HINDEFINITE, 1H,
1 4HSTEP, 1H, 18HBREAKPOINT, 1H, 18HPROGRAM RA, 3HNGE, 18HSCM DIRECT,
1 18H RANGE, 18HLCM DIRECT, 6H RANGE, 18HSCM BLOCK, 5HRANGE,
1 5HLCM BLOCK, 5HRANGE, 18HSCM PARITY, 1H, 18HSCM PARITY, 1H /
1 ITEST=AND(IARG,COMP(7777777))
15 IF(ITEST,NE,0) RETURN
16 ITEST=1
17 DO 10 I=1,12
18 ITROUB(I),=FALSE,
19 IF(AND(IARG,ITEST),NE,0) ITROUB(I)=,TRUE,
20 ITEST=2*ITEST
21 10 CONTINUE
22 DO 20 I=1,12
23 IF(,NOT,ITROUB(I)) GO TO 20
24 WRITE(NARG,30) ICAUSE(1,I),ICAUSE(2,I)
25 20 CONTINUE
26 RETURN
27 C
28 30 FORMAT(1H,51HTHE FOLLOWING CONDITION EXISTED AT THE TIME OF THE ,
29 1 7MABORT==2A10)
30 END

----------------------------------------------------------
SUBROUTINE MSHMKR

ROUTINE TO GENERATE THE INITIAL PROBLEM MESH AND SET THE
INITIAL QUANTITIES

ORIGINALLY WRITTEN BY A.A. AMBERN, LASL T-3
MODIFIED AND DOCUMENTED BY J.L. NORTON, LASL T-3, 1974

* ----- BEGIN COMDECK PARAM ----- *
COMMON/PDCOM/NSCP1, ITABP, ITABXP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLM5Z
*
* ----- END COMDECK PARAM ----- *

* ----- BEGIN COMDECK LCMAT ----- *
COMMON/YSC3/UFIRE(IPFB), EFIRE(IPFB), RHOFIR(IPFB), XFIRE(IPFB)
*
* ----- END COMDECK LCMAT ----- *

* ----- BEGIN COMDECK YSTORE ----- *
*
* ----- BEGIN COMDECK YAGID ----- *
*
* ----- BEGIN COMDECK YAQSC ----- *
*
* ----- BEGIN COMDECK YAQSC ----- *

DIMENSION X(1), XPAR(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM(1)
1 V(1), VG(1), R(1), SIE(1), MP(1), RMP(1), RCSQ(1), E(1), ETIL(1), RVOL
2 M(1), RM(1), VP(1), P(1), PL(1), UTL(1), UL(1), CO(1), VTL(1)
3 Y(1), ROL(1), AVXSV(1), AVYSV(1), DLSROI(1), DLRSQ(1), CAPGAM(1), TUG
4 SIG(1), TSB(1), GRROZ(1), GRROP(1), TUDVECC(1), MTIL(1)
5 CONC(1), CTEMP(1), ANCU(1), ANCV(1), GRSV(1), GZSV(1), X1SK(1), X24K(1)
6 Y1SK(1), Y24K(1), XR24K(1), DKL&M(1), AREA(1)
*
* ----- END COMDECK YAGID ----- *
*
* ----- BEGIN COMDECK YAGID ----- *

LOGICAL RE8TRT, FILMIPAPER, TURB
REAL LAM, MU
COMMON/YSC1/AA8C(9680)
COMMON/YSC1/AA8C(1)
*
* ----- BEGIN COMDECK YAGID ----- *

EQUIVALENCE(AASC(1), X, XPRA(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM(1)
1 V(1), VG(1), R(1), SIE(1), MP(1), RMP(1), RCSQ(1), E(1), ETIL(1), RVOL
2 M(1), RM(1), VP(1), P(1), PL(1), UTL(1), UL(1), CO(1), VTL(1)
3 Y(1), ROL(1), AVXSV(1), AVYSV(1), DLSROI(1), DLRSQ(1), CAPGAM(1), TUG
4 SIG(1), TSB(1), GRROZ(1), GRROP(1), TUDVECC(1), MTIL(1)
5 CONC(1), CTEMP(1), ANCU(1), ANCV(1), GRSV(1), GZSV(1), X1SK(1), X24K(1)
6 Y1SK(1), Y24K(1), XR24K(1), DKL&M(1), AREA(1)
REAL*8 MP,H0P,MPARP,MTIL

* START THERMODYNAMIC-COMPUTATION

SET UP THE NAMELIST INPUT TABLE

ASSIGN 530 TO IERRT
CALL TABDEF(MESH,4HNRAD,NR0D,IEFLAG,0,0,0,0)
CALL TABSET(MESH,4HNRAD,NR0D,IEFLAG,0,0,0,0)
CALL TABSET(MESH,6HFBFILE,FBFILE,IEFLAG,0,0,0,0)

INITIALIZE, NQIM IS THE NO. OF WORDS OF MEMORY NEEDED TO STORE
ONE ROW OF DATA (LESS 1).

CALL START
DO 50 J=2,JP2
  DO 40 I=1,IP1
  X(I,J)=XX
  Y(I,J)=YY
  R(I,J)=XXCYL+OMCYL
  U(I,J)=0,
  V(I,J)=0,
  IF(J,NE,2) GO TO 20
  Y(I,J)=YY+DZ
  X(I,J)=XX
  R(I,J)=R(I,J)
  U(I,J)=0,
  V(I,J)=0,
  IF(J,NE,JP2) GO TO 30
  Y(I,JP)=YY+DZ
  X(I,JP)=XX

50 CONTINUE
SEE IF THE ENTIRE MESH ACTUALLY IS UNIFORM

IF(FREZXR.EQ.1., AND, FREZYT.EQ.1., AND, FREZYM.EQ.1.,) GO TO 130

NO, DO THE NON-UNIFORM GENERATION,

CONVERT JCEN TO AN ACTUAL VERTEX NO, JCEN IS THE J-LINE THAT GOES THROUGH THE CENTER OF THE UNIFORM REGION,

JCEN=JCEN+2

JTOP AND JBOT ARE THE J-LINES AT THE TOP AND BOTTOM OF THE UNIFORM REGION

JTOP=JCEN+JUNF02
JBOT=JCEN-JUNF02

TJ IS THE DISTANCE FROM THE CENTER TO THE TOP (OR BOTTOM) OF THE UNIFORM REGION

TJ=FLOAT(JUNF02)*DZ

LOOP TO SET THE NON-UNIFORM VERTICES

CALL START

DO 110 J=2, JP2
DO 100 I=1, IP1
IMJ=I-JQ
IF(FREZXR.EQ.1.) GO TO 60

SEE IF WE ARE WITHIN THE UNIFORM X REGION

IF(I.LE.IUNF+1) GO TO 60

NO, GENERATE THE NON-UNIFORM X USING GEOMETRIC PROGRESSION,

X(IMJ)=X(IMJ)+FREZXR*(X(IMJ)-X(IMJ-NQ))
R(IMJ)=X(IMJ)*CYL+OMCYL

60 CONTINUE

JDT IS THE NO, OF J LINES THAT THE CURRENT J IS ABOVE THE TOP OF THE UNIFORM REGION

JDT=J-JTOP
C JDB IS THE NO. OF J LINES THAT THE CURRENT J IS BELOW THE BOTTOM OF THE UNIFORM REGION
C JDB=JDBOT*J
C SEE IF THE CURRENT J IS BELOW THE UNIFORM REGION
C IF(JDB,GT,0) GO TO 70
C NO, SEE IF IT IS ABOVE THE UNIFORM REGION.
C IF(JDT,GT,0) GO TO 80
C NO, J IS IN UNIFORM REGION, SET IT AS SUCH.
C Y(IJ)=REZY0+FLOAT(J=Jcen)*DZ
C GO TO 90
C 70 CONTINUE
C J IS BELOW THE UNIFORM REGION, CALCULATE ITS POSITION USING
C FORMULA FOR SUM OF GEOMETRIC PROGRESSION, THE FORMULA IS
C SUM=A*(1-F**N)/(1-F)
C WHERE A IS THE FIRST TERM, N IS THE NO. OF TERMS, AND F IS
C THE RATIO OF THE MTH TERM TO THE (M-1)TH TERM, IN THIS CASE
C A=DZ*FREZ, N=JDB, AND F=FREZ, THE POSITION OF JBOT IS
C AT Y=REZY0=TJ, J IS SUM BELOW THIS OR
C Y(IJ)=REZY0=TJ*DZ*FREZ*(1=FREZ**JDB)/(1-FREZ).
C 200 IF(FREZYB,GT,1.) Y(IJ)=REZY0=TJ=DZ*FREZYB*(1=FREZYB**JDB)*ROMFYB
C 201 GO TO 90
C 80 CONTINUE
C J IS ABOVE THE UNIFORM REGION, USE GEOMETRIC PROGRESSION SUM TO
C CALCULATE ITS VALUE ALSO.
C IF(FREZYT,GT,1.) Y(IJ)=REZY0+TJ+DZ*FREZYT*(1=FREZYT**JDT)*ROMFYT
C 90 CONTINUE
C SEE IF J IS 2
C IF(J,N.E.,2) GO TO 100
C YES, SET YB (BOTTOM Y).
C YB=Y(IJ)
C MAKE THE FICTITIOUS CELLS AT THE BOTTOM AS THOUGH THERE WERE
C ANOTHER REAL ROW OF CELLS
C X(IJM)=X(IJ)
C IJM=IJM+NO
C IJP=IJP+NO
C 100 IJ=IJ+NO
C CALL LOOP
C 110 CONTINUE
C CALL DONE
CALL START
DO 120 I=1,IP1
   Y(IJM)=Y(IJ)=(Y(IJP)+Y(IJ))*FREZYB
   IJM=IJ+IP
   IJP=IJP+IP
120 CONTINUE
CALL DONE

MESH IS GENERATED

130 CONTINUE

******************************************************************
GENERATE AN INITIAL FIREBALL AND ITS AMBIENT ATMOSPHERE
******************************************************************
FIRST SET UP THE AMBIENT ATMOSPHERE
GET GAMMA=1 AT REZY0
ETMP=REZSIE
RMTMP=REZRON
CNCJ=0
CALL AIR
XX IS THE ISOTHERMAL CONSTANT IN THE PRESSURE
XX*GMONE*REZSIE
NOTE THAT A NEGATIVE GRAVITATIONAL FORCE IS ASSUMED HERE
YY=5*ABS(GZ)
BRING IN THE FIRST THREE ROWS
CALL START
PROCESS J=1 AND 2 ROWS, YJC2 IS THE Y COORDINATE OF CENTERS OF
   CELLS IN THE ROW J=2
YJC2=5*(Y(IJP)+Y(IJ))
ROSAY IS THE DENSITY OF THE CELLS IN ROW J=2 ASSUMING AN
   ISOTHERMAL, IDEAL GAB ATMOSPHERE
ROSAY=REZRON*EXP(-GZ*(REZY0-YJC2)/XX)
CALCULATE THE DENSITY OF CELLS IN ROW J=1 USING THE DIFFERENCE
   FORM FOR HYDROSTATIC EQUILIBRIUM
FNUM=(Y(IJP)+Y(IJ))*YY
FDEN=FNUM/FREZYB
ROJ1=ROSAY*(XX+FNUM)/(XX*FDEN)
C SET AMBIENT QUANTITIES IN ROWS J=1 AND 2

DO 140 I=1,IP1
    RO(IJ)=ROAV
    RO(IJM)=ROJ1
    E(IJ)=REZSIE
    SIE(IJ)=REZSIE
    E(IJM)=REZSIE
    SIE(IJM)=REZSIE
    IJ=IJ+1
140 IJM=IJM+NQ

C BRING IN THE NEXT ROW
C CALL LOOP
C LOOP OVER ALL THE OTHER REAL ROWS, BOOTSTRAP DENSITIES UPWARD FROM ROW J=2.

DO 160 J=3,JP1
    FDEN=FDEN*FREZYT
    FNUM=FNUM*FREZYT
    ROJPZ=ROAV*(XX+FNUM)/(XX+FDEN)
    DO 150 I=1,IP1
        RO(IJ)=ROAV
        E(IJ)=REZSIE
        SIE(IJ)=REZSIE
        IJ=IJ+NQ
    150 CONTINUE
C SET THE TOP FICTIONAL ROW
C FNUM=FNUM*FREZYT
C FDEN=FDEN*FREZYT
C ROJPZ=ROAV*(XX+FNUM)/(XX+FDEN)
C DO 170 I=1,IP1
C RO(IJ)=ROJPZ
C E(IJ)=REZSIE
C SIE(IJ)=REZSIE
170 IJ=IJ+NQ
C CALL DONE

C THE AMBIENT ATMOSPHERE IS NOW SET, READ IN THE DATA TO GENERATE THE FIREBALL, ISUB IS THE DATA POINT SUBSCRIPT, INTERPOLATION TABLES ARE READ STARTING WITH ENTRY 2.

ISUB=1

C READ INFORMATION ABOUT THE FINENESS OF THE FIREBALL INTERPOLATION AND FROM WHERE THE FIREBALL INPUT DATA WILL COME

IBFILE=5
NRAD=5
NTH=180
FBFILE=.FALSE.,
CALL NAMLST(MESH,5,IEFLAG)
IF(IEFLAG,.NE.,0) CALL UNCLE(4,6,HMSHMKR,25,
1 25HFIRE NAMELIST INPUT ERROR)
C
C SEE IF INPUT WILL BE FROM SPECIAL FILE
C
IF(.NOT.,FBFILE) GO TO 200
C
YES, PRINT A MESSAGE TELLING THE USER SO.
C
DO 180 IPX=6,IFD,6
180 WRITE(IPX,540)
FBFILE=
C
PRINT THE DATA TABLE HEADER
C
DO 190 IPX=IPD,IFD,6
190 WRITE(IPX,570)
C
LOOP TO READ DATA POINTS
C
200 CONTINUE
ISUB=ISUB+1
C
SEE IF NEXT READ WILL OVERFLOW THE TEMPORARY STORAGE
C
IF(ISUB.GT.,IPFB) CALL UNCLE(4,6,HMSHMKR,29,
1 29HToo MANY FIREBALL INPUT CARDS)
C
FIREBALL DATA IS IN SPHERICAL LAGRANGIAN FORM, INPUT IS
PRESSURE,RADIAL VELOCITY,SPECIFIC INTERNAL ENERGY,AND
DENSITY, IN CGS UNITS, THE VELOCITY IS DEFINED AT THE GIVEN
RADIUS, THE SPECIFIC INTERNAL ENERGY AND DENSITY FOR THE
CELL BETWEEN XFIRE(K=1) AND XFIRE(K) IS READ IN WITH
XFIRE(K). THE FIREBALL IS ASSUMED TO BE
CENTERED AT REZY0. THE PRESSURE IS ONLY INFORMATIVE, IT IS
NOT USED IN THE CALCULATION BUT RATHER IS RECALCULATED FROM
THE INPUT ENERGY AND DENSITY USING THE EQN-OF-STATE,
C
READ(IBFILE,580) PJLN,UFIRE(ISUB),EFIRE(ISUB),RHOFIR(ISUB),XFIRE(
1 ISUB)
C
CHECK FOR EOF ON INPUT, IF SO, CONSIDER END OF INPUT.
C
IF(EOF,IBFILE) 230,210
C
210 CONTINUE
C
CALCULATE THE PRESSURE FROM THE EQN-OF-STATE TO BE USED IN THE
PRINTOUT TO COMPARE WITH THE INPUT PRESSURE, A ZERO DENSITY
INDICATES THAT ALL THE DATA CARDS HAVE BEEN READ, (A BLANK
CARD TERMINATES THE INPUT.)
C
RHOF=RHOFIR(ISUB)
C
IF(RHOF,EQ.,0.) GO TO 230
C
GO GET GAMMA=1 FROM THE EOS. IF SIE,GE,1.E10, METHANE IS
ASSUMED (CONCJ=1.), OTHERWISE, AIR IS USED (CONCJ=0.).

ETMP=EFIRE(ISUB)
CONCJ=0.
IF(ETMP, GE, 1.E10) CONCJ=1,
CALL AIR
PRHO=GMONE*ROTMP*ETMP
PRINT OUT THE INPUT DATA AND THE COMPARISON PRESSURES FOR
THIS DATA POINT
DO 220 IPX=IPD, IFD, 6
  220 WRITE(IPX,590) UFIRE(ISUB),EFIRE(ISUB),RHOFIR(ISUB),XFIRE(ISUB),
    1 PJLN,PRHO
SEE IF RADIUS IS MONOTONIC INCREASING UNLESS FIRST DATA POINT,
IN THE LATTER CASE, GO ON AND PROCESS THE NEXT POINT,
IF(ISUB, EQ, 2) GO TO 200
IF(XFIRE(ISUB), LE, XFIRE(ISUB-1)) CALL UNCLE(4,6HMSHMKR,40,
1 40HINPUT RADIUS ARE NOT MONOTONIC INCREASING)
GO BACK AND PROCESS THE NEXT POINT
GO TO 200
CONTINUE
DO 240 IPX=IPD, IFD, 6
  240 WRITE(IPX,550)
GET RID OF THE FIREBALL INPUT DATA FILE IF ONE EXISTS
IF(FBPILE) CALL CLO41T(3)
ALL DONE READING INPUT, SET THE DATA POINT COUNT TO THE TRUE NO,
ISUB=ISUB-1
SET THE QUANTITIES AT THE CENTER OF THE SPHERE, THE ENERGY AND
DENSITY ARE SET TO THOSE AT THE FIRST DATA POINT READ, THE
VELOCITY IS ZERO, OF COURSE,
UFIRE(1)=0,
EFIRE(1)=EFIRE(2)
RHOFIR(1)=RHOFIR(2)
XFIRE(1)=0.
LOOP OVER ALL ZONES AND INTERPOLATE THE CELL QUANTITIES
CALL START
DO 300 J=2, JP1
  300 DO 290 I=1, IBAR
  290 WRITE(399,*)
  300
  290
  280
  270
  260
  250
  240
  230
  220
  210
  200
  190
  180
  170
  160
  150
  140
  130
  120
  110
  100
  90
  80
  70
  60
  50
  40
  30
  20
  10
  0

FIND THE VERTEICES OF THE CELL OF INTEREST
VXL=V(IJ)
VXK=V(IPJ)
VXR=X(IPJ)
CONC(IJ) = 0, INDICATES NOTHING HAS BEEN SET IN THE ZONE YET.

LOOP OVER ALL 1-D ZONES

DO 280 K=2, ISUB

SEE IF ZONE COULD NOT POSSIBLY FALL WITHIN THE YAQUI ZONE

RVRT=VXR+(VYT-REZYO)**2
RVRB=VXR+(VYB-REZYO)**2
RFIRE=RFIRE(K-1)**2
IF(RFIRE > RVRT AND RFIRE > RVRB) GO TO 280
RVLT=VXL+VXR+(VYT-REZYO)**2
RVLB=VXL+VXR+(VYB-REZYO)**2
RFIRE=RFIRE(K)**2
IF(RFIRE < RVLT AND RFIRE < RVLB) GO TO 280

LOOP OVER ANGULAR INCREMENTS

DO 270 IT=1, NTH
THETAL=THETA2
THETA2=THETA2+DNTH
CTHETA=6*(THETAL+THETA2)
COST=COST(CTHETA)

LOOP OVER RADIAL INCREMENTS

DO 240 I=1, NRAD
R1=R2
R2=R2+DR
CR=5*(R1+R2)
XCEN=CR*SINT

SEE IF THE X-COORDINATE OF THE CENTER OF THE LAGRANGIAN PIECE LIES IN THE YAQUI CELL OF INTEREST

IF(XCEN < VXU OR XCEN > VXR) GO TO 260

YES, CHECK THE Y-COORDINATE

YCEN=CR*COST+REZYO
IF(YCEN < VYU OR YCEN > VYT) GO TO 260

CONSIDER THE PIECE IN THE YAQUI CELL, SEE IF ANY PIECES HAVE PREVIOUSLY BEEN FOUND.
IF(CONC(IJ),NE.0.,) GO TO 250
C
NO, INITIALIZE;
C
RO(IJ)=0.
U(IJ)=0.
V(IJ)=0.
$IE(IJ)=0.
250 CONTINUE
C
COMPUTE THE MASS OF THE PIECE, ACCUMULATE THE VOLUME
IN CONC,
C
CONC(IJ)=CONC(IJ)+VOL
XMASS=VOL*RHOFIR(K)
C
COMPUTE THE VELOCITIES OF THE PIECE
C
RATIO=CR*FIRE(K-1))/XFIRE(K-1))
ROOTUFIRE(K)*RATIO=UFIRE(K-1))
XDOT=ROOT*INT
YDOT=ROOT*COST
C
DEPOSIT MASS, MOMENTUM, AND ENERGY IN THE YAQUI ZONE
C
U(IJ)=U(IJ)+XDOT*XMASS
V(IJ)=V(IJ)+YDOT*XMASS
$IE(IJ)=SIE(IJ)+EFIRE(K)*XMASS
RO(IJ)=RO(IJ)+XMASS
260 CONTINUE
270 CONTINUE
280 CONTINUE
290 CONTINUE
CALL LOOP
300 CONTINUE
CALL DONE
C
LOOP OVER ALL CELLS AND CHECK FOR FIREBALL CELLS
C
CALL START
DO 390 J=2,JPI
DO 380 I=1,IBAR
IF(CONC(IJ),.EQ.,0.,) GO TO 370
C
FIREBALL CELL FOUND, SEE IF ITS VOLUME IS VERY FAR OFF,
C
IPJ=IJ+Q
VOL=.5*(Y(IJP)-Y(IJ))*X(IPJ)**2=X(IJ)**2)
RERROR=(CONC(IJ)-VOL)/VOL
IF(ABS(RERROR).LT.,.01) GO TO 370
C
ERROR IS LARGE, INFORM THE USER AND CONSIDER CELL AS
FIREBALL BOUNDARY CELL.
DO 310 IPX=6,IFD,6
310 WRITE(IPX,560) I,J,ERROR
C USE VALUES FROM NEIGHBORING CELL FOR DETERMINING
CONTRIBUTION OF PART OF CELL OUTSIDE OF THE FIREBALL
C
IF(CONC(IPJ),NE;0,0) GO TO 320
C
USE CELL TO RIGHT
INDEX=IPJ
GO TO 360
320 CONTINUE
IF(CONC(IJP),NE;0,0) GO TO 330
C
USE CELL ABOVE
INDEX=IJP
GO TO 360
330 CONTINUE
IF(CONC(IJM),NE;0,0) GO TO 340
C
USE CELL BELOW
INDEX=IJM
GO TO 360
340 CONTINUE
IPJ=IJP+NQ
IF(CONC(IPJP),NE;0,0) GO TO 350
C
USE CELL TO UPPER RIGHT
INDEX=IPJP
GO TO 360
350 CONTINUE
IJM=IJM+NQ
IF(CONC(IJM),NE;0,0) CALL UNCLE(I,6HMSHMKR,5B,
1 58ERROR IN PROCESSING CELL WITH LARGE RELATIVE ERROR)
C
USE CELL TO LOWER RIGHT
INDEX=IJM
360 CONTINUE
C
ADD IN CONTRIBUTION FROM PART OF CELL OUTSIDE OF THE FIREBALL
C
DV=VOL*CONC(IJ)
DM=DV/RO(INDEX)
RO(IJ)=RO(IJ)+DM
SIE(IJ)=SIE(IJ)+DM*SIE(INDEX)
370 CONTINUE
IJP=IJP+NQ
IJM=IJM+NQ
380 CONTINUE
CALL LOOP
390 CONTINUE
ALL FIREBALL CELLS ARE NOW COMPLETELY DEFINED.
GO BACK AND DEFINE THE VELOCITIES, STORE THEM TEMPORARILY IN UG,VG.

CALL START
DO 440 J=2,JP1
DO 430 IM1,IBAR
UG(IJ)=0., VG(IJ)=0., IPJ=IJ+NQ

SEE IF THE CURRENT CELL IS A FIREBALL CELL
IF(CONC(IJ),EQ,0.) GO TO 420
YES, ALL NEIGHBORING CELLS OF THE LOWER LEFTHAND VERTEX MUST ALSO BE FIREBALL CELLS OR VELOCITY IS SET TO ZERO.
IF(I,EQ,1) GO TO 400
IMJ=IJ-NQ IF(CONC(IMJ),EQ,0.) GO TO 420
IMJM=IJM-NQ IF(CONC(IMJM),EQ,0.) GO TO 420
400 CONTINUE
IF(CONC(IJM),EQ,0.) GO TO 420
ALL NEIGHBORS ARE FIREBALL CELLS, DIVIDE UP THE MOMENTUM.
XMASS*RO(IJ)+RO(IJM)
UG(IJ)=U(IJ)+U(IJM)
VG(IJ)=V(IJ)+V(IJM)
IF(I,EQ,1) GO TO 410
IMJ=IJ-NQ
IMJM=IJM-NQ
XMASS*XMASS*RO(IMJ)+RO(IMJM)
UG(IJ)=UG(IJ)+U(IMJ)+U(IMJM)
VG(IJ)=VG(IJ)+V(IMJ)+V(IMJM)
410 CONTINUE
UG(IJ)=UG(IJ)/XMASS
VG(IJ)=VG(IJ)/XMASS
420 CONTINUE
IJ=IJ+NP
IJP=IJP+NP
IJM=IJM+NP
430 CONTINUE
CALL LOOP
440 CONTINUE
CALL DONE

NOW WE HAVE VERTEX VELOCITIES IN UG,VG, STORE THEM IN U,V AND CONVERT MASSES TO DENSITIES AND INTERNAL ENERGIES TO SPECIFIC INTERNAL ENERGIES.

CALL START
DO 470 J=2,JP1
DO 460 IM1,IBAR
IF(CONC(IJ),EQ:O:) GO TO 450

U(IJ)=UG(IJ)
V(IJ)=VG(IJ)

IPJ=IPJ+NQ
VOLX=V(X(IJP)-Y(IJP))*(X(IPJ)**2-X(IJ)**2)
SIE(IJ)*SIE(IJ)/RO(IJ)
RO(IJ)*RO(IJ)/VOL

CONTINUE

IJ=IJ+NQ
IPJ=IPJ+NQ
CONTINUE

CALL LOOP
CONTINUE
CALL DONE

SET CONCENTRATIONS, CONCJ=1, INDICATES METHANE, CONCJ=0, IS AIR.
EVERYTHING WITH SIE,GE,1,E10 IS CONSIDERED METHANE, ALL ELSE IS AIR.

CALL START

SET THE BOTTOM FICTITIOUS ROW
DO 480 I=1,IP1
CONC(IJM)=0.
IJM=IJM+NQ
CONTINUE

SET THE REAL ROWS AND THE RIGHT FICTITIOUS COLUMN
DO 500 J=1,JP1
DO 490 I=1,IP1
CONC(IJ)=0.
IF(SIE(IJ),GE,1,E10,AND, I,NE,IP1) CONC(IJ)=1,
IJ=IJ+NQ
490 CONTINUE
CALL LOOP
CONTINUE

SET THE TOP FICTITIOUS ROW
DO 510 I=1,IP1
CONC(IJ)=0.
IJ=IJ+NQ
510 CONTINUE
CALL DONE

ZERO OUT U AND V IN THE BOTTOM FICTITIOUS ROW
CALL START
DO 520 I=1,IP1
U(IJM)=0.
V(IJM)=0.
IJM=IJM+NQ
520 CONTINUE
CALL DONE
RETURN
SUBROUTINE MTHANE

ROUTINE TO CALCULATE THE FACTOR GAMMA-1 FOR METHANE

UNITs ARE ALL CGS

ORIGINALLY OBTAINED FROM THE AIR FORCE WEAPONS LAB
MODIFIED BY J.L. NORTON, LASL T=3, 1974

COMMON/CONST/RH0,E,GHONE
DIMENSION A(8)
DATA A/3.21782E-1,1.56848E-4,8.78899E-2,1.25271E-3,1.46612E-2,
1.72468E-6,7.72322E-4,6.63413E-6/
RETURN
END

SUBROUTINE NAMLST(TABLE, IF, IEFLAG)

ROUTINE TO REPLACE COMPILER/SYSTEM DEPENDENT NAMELIST INPUT,
HERE, AN ATTEMPT HAS BEEN MADE TO ELIMINATE AS MANY
COMPILER- AND/OR SYSTEM-DEPENDENT FEATURES AS POSSIBLE, THOSE
STILL REMAINING ARE BOOLEAN ALGEBRA, 60 BIT WORD ASSUMPTION,
INLINE SHIFT FUNCTION (CAN BE REPLACED BY ASSEMBLY-LANGUAGE
FUNCTION IF IT IS TYPED INTEGER), AND TWO ASSEMBLY-LANGUAGE
ROUTINES TO FETCH AND STORE WORDS FROM AND INTO ABSOLUTE
MEMORY LOCATIONS.

TABLE = NAMELIST DATA TABLE
IF = INPUT FILESET
IEFLAG = ERROR FLAG
    = 0 = NO ERROR
    = -1 = END-OF-FILE ON FILESET IF
    = N (.NE.0 OR =1) = ERROR EXIT
    (ERROR MESSAGE WILL BE PRINTED BY MESSOT)
    (IF .LT.0, EOF ALSO OCCURRED)

LIMITATIONS = CURRENTLY, ONLY SMALL CORE VARIABLES MAY BE READ.
ARRAYS OF UP TO FOUR SUBSCRIPTS CAN BE INPUT ALTHOUGH MANY
COMPILERS CAN HANDLE A MAXIMUM OF ONLY THREE SUBSCRIPTS.

INPUT RULES = DATA MAY BE INPUT IN ANY OF THREE GENERAL FORMS ==

V=D1,D2,...,DN
A(D1,D2,...,DN)

WHERE V IS A NON-SUBSCRIPTED VARIABLE,
A IS AN ARRAY,
SUB REPRESENTS FROM 1 TO 4 SUBSCRIPTS (INTEGER CONSTANT)
AND DN REPRESENTS A DATA ELEMENT.
THE DATA ELEMENTS MAY BE OF A NUMBER OF TYPES ==
INTEGER = BASE 10 (EXAMPLE = 13569)
    = BASE 8 (FOLLOW CONSTANT WITH B) (7765B)
REAL = FIXED (13,25)
    = FLOATING (3,265E29)
COMPLEX (= -19,59,3,14E-7)
DOUBLE = 3.141592653687962E3846D3
LOGICAL = T, F
MOLLERITH = LEFT=JUSTIFIED, BLANK=fill (5THING)
            = LEFT=JUSTIFIED, ZERO=fill (3LOUT)
            = RIGHT=JUSTIFIED, ZERO=fill (GRWORD)
MULTIPLIERS ARE ALLOWED, BUT ONLY FOR ONE ELEMENT AT A TIME,
FOR EXAMPLE, 5*23, 6*15, 3*21 = 27, 3E7, 5*4HE8 TEST ARE ALL
LEGAL, 6*(13, 21, 7) WOULD WORK, BUT ONLY BECAUSE THE
QUANTITY INSIDE THE PARENTHESES IS INTERPRETED AS A
SINGLE COMPLEX CONSTANT, 22*(1,2,3,4,5,6) WOULD BE
FLAGGED AS AN ERROR, 5*13ABCDEFHJKLM IS ALSO ILLEGAL
AS THE CONSTANT IS MORE THAN ONE WORD LONG,
INTEGERS WITH MORE THAN 14 DIGITS AND REAL CONSTANTS OF
MORE THAN 14 DIGITS NOT INDICATED AS DOUBLE PRECISION
ARE ILLEGAL. FOR EXAMPLE, 3.141592653897932 IS WRONG
BUT PI=3.1415926536879932D 18 OK.

NOTE CAREFULLY = NO CONVERSIONS OF DATA TYPE ARE MADE, THIS GIVES
ONE THE CAPABILITY OF STORING REAL CONSTANTS INTO INTEGER
VARIABLES, FOR EXAMPLE, HOWEVER, IF ONE FORGETS TO PUNCH A
DECIMAL POINT (.5=5), THE CONSTANT WILL BE STORED AS AN
INTEGER.

OTHER CONVENTIONS AND CAPABILITIES ARE AS FOLLOWS ==
ANY CARD WITH CS IN CC1=2 WILL BE PRINTED OUT AND IGNORED,
ANY CARD WITH PS IN CC1=2 WILL TURN ON A PRINT SWITCH AND
EACH CARD OF THE NAMELIST INPUT RECORD (UP TO THE NEXT S)
WILL BE PRINTED BEFORE PROCESSING,
AN ISOLATED = IS ILLEGAL, =EP=1,EB=, DB=1, DB=, =.TRUE., =.FALSE.
AND = FALSE, = TRUE, = B = 0A. A MINUS SIGN IN FRONT OF AN
OCTAL CONSTANT CAUSES THE CONSTANT TO BE COMPLEMENTED.
A MINUS SIGN IN FRONT OF A HOLLERITH CONSTANT IS IGNORED.
AN R HOLLERITH CONSTANT CANNOT BE MORE THAN 10 CHARACTERS
LONG.
THE TERMINATION $ MUST NOT OCCUR IN CC1 OR CC2. IF IT OCCURS
IN CC1, IT WILL BE IGNORED. AN ERROR WILL BE RETURNED IF
IT IS IN CC2.
A NAMELIST RECORD MAY BE OF ANY LENGTH, BLANKS ARE SIGNIFICANT
AND MAY OCCUR ONLY AROUND NON-NUMERIC CHARACTERS OTHER THAN
EXPONENTS, FOR EXAMPLE, LEGAL BLANKS ARE X = 5,
Y = 10.3, L = TRUE, C = ( 5, 6 ), I = 15.
ILLEGAL BLANKS WOULD INCLUDE X = 5, E = 10, L = TRUE, Y = 5.
THE NAMELIST RECORD IS TREATED AS ONE LONG STRING OF
CHARACTERS, ALL 80 CHARACTERS OF EACH CARD BEING SCANNED,
EXCEPT THAT CC2 OF EACH CARD MAY NOT BE A $ UNLESS IT IS
PART OF A HOLLERITH FIELD.
BECAUSE OF THE UNLIMITED LENGTH OF A NAMELIST RECORD, A
HOLLERITH CONSTANT MAY BE OF UNLIMITED LENGTH.
BECAUSE TYPE IS NOT CHECKED, STORING A DOUBLE-PRECISION
CONSTANT INTO A SINGLE-PRECISION VARIABLE WILL CLOBBER
THE FOLLOWING LOCATION, (X = 5.0) WILL STORE A ZERO INTO
THE LOCATION FOLLOWING X ASSUMING X IS NOT DOUBLE OR
COMPLEX), A SIMILAR WARNING CAN BE GIVEN FOR X = (5, 2).
**********w**********k*************************************************
**EXAMPLE OF USAGE**
**SUPPOSE ONE WISHED TO REPLACE THE STATEMENTS**
DIMENSION A(20), B(5, 3), X(5, 10, 15)
NAMELIST/CARDN/I, J, K, A, B, X
READ(5, CARDN)
WITH SYSTEM-INDEPENDENT INPUT. FURTHERMORE, SUPPOSE ONE WISHED
TO HAVE PS AND CS CARDS COME OUT ON BOTH PAPER AND FILM
AND ERROR MESSAGES COME OUT ON FILE 59, THEN ONE WOULD
 NEED THE FOLLOWING =
DIMENSION A(20), B(5, 3), X(5, 10, 15), ITAB(2), TABLE(3, 7)
C SET UP FILE TABLE FOR MESSAGES
DATA ITAB/3, 4, TABLE/5, 6, 7,
CALL NAMPRT(2, ITAB)
CALL ERRPRT(1, ITAB)
C DEFINE THE NAMELIST TABLE, IF THERE ARE TO BE N UNIQUE
INPUT VARIABLE NAMES, THEN 3* (N + 1) TABLE LOCATIONS
ARE NEEDED, THUS, FOR SIX VARIABLES 3* 7 LOCATIONS
MUST BE SET ASIDE. THE THIRD ARGUMENT IS THE
SECOND SUBSCRIPT OF TABLE, IF ANY ERRORS OCCUR IN
SUBSEQUENT CALLS TO TABSET, TRANSFER WILL OCCUR TO
THE STATEMENT NO. ASSIGNED TO IERRT. THIS ELIMINATES
ERROR CHECKING AFTER EACH CALL TO TABSET. IFLAG CAN
THEN BE EXAMINED TO DETERMINE WHAT TYPE OF ERROR
OCCURRED.
ASSIGN 999 TO IERRT
CALL TABDEF(TABLE,5HCARDN,7,IERRT)
ENTER EACH VARIABLE INTO THE TABLE, IF THERE IS SOME
ERROR, IEFALG IS RETURNED NON-ZERO,
CALL TABSET(TABLE,1HI,I,IEFLAG,0,0,0,0)
(C THE 0 DENOTES A NON-SUBSCRIPTED VARIABLE)
CALL TABSET(TABLE,1HI,I,IEFLAG,0,0,0,0)
CALL TABSET(TABLE,1HI,K,IEFLAG,0,0,0,0)
CALL TABSET(TABLE,1HA,A,IEFLAG,1,0,0,0)
(THE 1 DENOTES A SUBSCRIPTED VARIABLE BUT THE SUBSCRIPT
NEED NOT BE GIVEN FOR A SINGLE-SUBSCRIPTED VARIABLE)
CALL TABSET(TABLE,1HB,B,IEFLAG,2,5,0,0)
(THE VARIABLE IS DOUBLE-SUBSCRIPTED BUT ONLY THE FIRST
SUBSCRIPT NEED BE GIVEN)
CALL TABSET(TABLE,1HX,X,IEFLAG,3,5,10,0)
(THE VARIABLE IS TRIPLY-SUBSCRIPTED BUT ONLY THE FIRST
TWO SUBSCRIPTS NEED BE GIVEN)
NOTE THAT TABSET IS CALLED WITH THE FULL NO. OF ARGUMENTS
WHETHER THEY ARE USED OR NOT. THIS IS NOT STRICTLY
NECESSARY ON MOST COMPILERS; CHECK YOUR LOCAL
CONVENTIONS.
READ IN A NAMELIST CARD, IF IEFALG RETURNS 0, ALL WENT OK.
IF IEFALG RETURNS -1, A NORMAL EOF WAS ENCOUNTERED,
FOR ALL OTHER VALUES OF IEFALG, AN ERROR OCCURRED,
IF IEFALG,LT,0, AN EOF ALSO OCCURRED.
CALL NAMELIST(TABLE,5,IEFLAG)
(5 IS THE INPUT FILE NO.)
IF(IEFLAG, EQ, 0) GO TO 10
IF(IEFLAG, EQ, (-1)) GO TO 20
A NAMELIST ERROR OCCURRED
STOP 7
A NAMELIST TABLE INITIALIZATION ERROR OCCURRED
999 CONTINUE
. . .
AN EOF OCCURRED
20 CONTINUE
NORMAL EXIT
10 CONTINUE

*********:*******************************

WRITTEN BY J. L. NORTON

THESE VERSION RUNS ON A CDC 6600 OR 7600 USING THE RUN COMPILER

DIMENSION TABLE(3,1), IDSTR(80)
INTEGER TABLE, AND, OR, COMP, SHIFT
LOGICAL LIST, CMOPLEX, DIM, DELIM, FIX, MULT, CST, HOLLER, DOBLE
LOGICAL $AVE, FIRST, DONE, EXPN, PERIOD, LCONT, LCONF, CDONE, CPFND
COMMON/ARRAY/ICICHAR(80), ISUB, ITEST, NSUB, NSUBV(4), IENTRY, ISSBPT,
CREAL, CIMAG, CMPLX, DIM, MULTSV, IVSUM, MULT, LIST, HOLLER, DOBLE
COMMON/ERROR/IERROR
COMMON/SHIFT/ XLEFT, XRIGHT, XDUM
DOUBLE PRECISION DB, DBP, DCONST, DTEMP, DZERO
EQUIVALENCE (DB, CREAL)
EQUIVALENCE (IVSUM, XVUM)
DATA LCONT, LCONF, TRUE, FALSE, /
DATA IAF, IAI, IRA, IRZ/
DATA INF, INL, IRG, IFR/
DATA DCONST/10.,0,1,DZERO/0,0,
DATA IEMIN, IEMAX/-294,322/
DATA NOSP, NDDP/14, 26/
DATA INAME/0/

LIST=IF TRUE, EACH NAMELIST CARD INPUT WILL BE LISTED
LIST=FALSE,
BEGINNING OF CODE FOR PROCESSING A NAMELIST RECORD
10 CONTINUE

RESET THE ERROR FLAG
IEFLAG=0
READ A CARD
CALL READIT(IF, IFLAG)
IF AN END-OF-FILE WAS ENCOUNTERED, QUIT
IF (IEFLAG, LT, 0) RETURN
ISUB IS THE POINTER TO THE COLUMN BEING PROCESSED OF THE CARD
LAST READ, BEGIN LOOKING AT CC2
ISUB=2
IF CC2 IS NOT A DOLLAR, IGNORE THE CARD AND GO BACK AND READ
ANOTHER ONE

IF(ICHAR(2),NE,'1H$) GO TO 10

CC2 WAS A DOLLAR, IF CC1 IS A -=P-, TURN ON THE LIST FLAG,
IF(ICHAR(1),EQ,'1MP) LIST=.TRUE.

IF CC1 IS A C, PRINT THE CARD AND GO READ ANOTHER
IF(ICHAR(1),NE,'1MC) GO TO 20
CALL MESS0T(1)
GO TO 10

CARD IS TO BE PROCESSED.

CONTINUE

IVNUM IS THE NO. OF CHARACTERS READ IN THE CURRENT STRING
IVNUM=0

*************************************************************************
DECODE THE NAMELIST NAME
*************************************************************************

30 CONTINUE

GO TO THE NEXT COLUMN
ISUB=ISUB+1

ITEMP IS THE CHARACTER IN CC ISUB, LEFT-JUSTIFIED, BLANK FILL
ITEMP=ISHIFT(ITEMP,-54)
CHECK FOR THE FIRST CHARACTER OF THE STRING
IF(ITEMP,EQ,'1H ) GO TO 50
NO. UP THE CHARACTER COUNT, IF MORE THAN SIX, FATAL ERROR,
IVNUM=IVNUM+1
IF(IVNUM,LE,6) GO TO 40
CALL MESS0T(2)
IFLAG=2
RETURN

40 CONTINUE
ITEMP IS THE CHARACTER IN CC ISUB, RIGHT-JUSTIFIED, ZERO FILL
ITEMP=AND(SHIFT(ITEMP,-54),778)
CHECK FOR THE FIRST CHARACTER OF THE STRING
IF(IVNUM,GT,1) GO TO 50
298 C YES, IF NON-ALPHABETIC, FATAL ERROR.
299 C
300 C IF (ITEMP .GE. IAF, AND, ITEMP .LE. IAL) GO TO 60
301 C CALL MESSOT (23)
302 C IFLAG = 23
303 C RETURN
304 C 50 CONTINUE
305 C
306 C IF ANY CHARACTER IS NOT A LETTER OR A NUMBER, FATAL ERROR.
307 C
308 C IF ((ITEMP .GE. INF, AND, ITEMP .LE. INLI, OR, (ITEMP .GE. IAF, AND, ITEMP .LE.
309 C 1 IAL)) GO TO 60
310 C CALL MESSOT (24)
311 C IFLAG = 24
312 C RETURN
313 C 60 CONTINUE
314 C
315 C STORE THE CHARACTER AND GO LOOK AT THE NEXT
316 C
317 C IDSTR (IVNUM) = ITEST
318 C GO TO 30
319 C
320 C A BLANK CHARACTER HAS BEEN FOUND
321 C
322 C 70 CONTINUE
323 C
324 C IF IT IS THE FIRST CHARACTER AFTER THE $, FATAL ERROR
325 C
326 C IF (IVNUM .GT. 0) GO TO 80
327 C CALL MESSOT (11)
328 C IFLAG = 11
329 C RETURN
330 C 80 CONTINUE
331 C
332 C GET THE STRING INTO ONE WORD (INAME), LEFT-JUSTIFIED, BLANK FILL
333 C
334 C CALL MASH (IDSTR, IVNUM, INAME, IFLAG)
335 C IF (IFLAG .NE. 0) RETURN
336 C
337 C SEE IF IT MATCHES THE NAME IN THE NAMELIST TABLE, IF NOT, GO BACK
338 C AND READ THE NEXT CARD.
339 C
340 C IF (TABLE (1, 1) .NE. INAME) GO TO 10
341 C
342 C NAME MATCHES, INE IS THE NUMBER OF VARIABLES REPRESENTED IN THE
343 C TABLE + 1. DELIM = FALSE, INDICATES THAT A DELIMITER ($ or $)
344 C IS NOT EXPECTED AS THE NEXT CHARACTER, FIRST = TRUE, INDICATES
345 C THAT A VARIABLE NAME MUST BE FOUND BEFORE A CONSTANT.
346 C
347 C INE = TABLE (3, 1) + 1
348 C DELIM = FALSE.
349 C FIRST = TRUE.
350 C
351 C LIST THE CARD IF THE FLAG IS ON
352 C
353 C IF (LIST) CALL MESSOT (1)
354 C GO TO 100
MAIN LOOP FOR DECODING VARIABLE NAMES AND CONSTANTS

90 CONTINUE

ONE VARIABLE NAME AND CONSTANT HAVE ALREADY BEEN PROCESSED,
 EITHER A VARIABLE OR A CONSTANT CAN OCCUR NEXT,

FIRST=.FALSE.,

SUBLOOP TO DETERMINE WHETHER NEXT STRING IS VARIABLE NAME OR
 CONSTANT

100 CONTINUE

NXTCOL PUTS THE CONTENTS OF THE NEXT CARD COLUMN INTO ITES$ALSO
 UPDATING ISSUE, IF NECESSARY, ANOTHER CARD IS READ AND ERROR
 CHECKS ARE PERFORMED,

CALL NXTCOL(IF,IEFLAG)

IF ANY ERRORS, QUIT RIGHT HERE

IF(IEFLAG,NE,0) RETURN

NO ERRORS, SEE IF NEXT CHARACTER IS A BLANK, IF IT IS, GO ON TO
 THE NEXT COLUMN,

IF(ITEMP,NE,1H ) GO TO 100

NOT A BLANK, SEE IF IT IS A DOLLAR, IF SO, WE ARE ALL DONE,

IF(ITEMP,NE,1HS) RETURN

NOT A DOLLAR, SEE IF A DELIMITER IS EXPECTED,

IF(.NOT.,DELIM) GO TO 110

YES, IF CHARACTER IS NOT A COMMA, FATAL ERROR,

IF(ITEMP,NE,1Hs) GO TO 260

CHARACTER IS A COMMA, REMOVE DELIMITER FLAG AND GO EXAMINE
 THE NEXT COLUMN,

DELIM=.FALSE.,

GO TO 90

110 CONTINUE

SEE IF CHARACTER IS ALPHABETIC

ITEMP=AND(SHIFT(ITEMP,=54),$7B)

IF(ITEMP,GE,IAF,AND,ITEMP,LE,IAL) GO TO 130
NO, IF A NAME IS EXPECTED, FATAL ERROR.
IF (FIRST) GO TO 120
A NAME IS NOT NECESSARILY EXPECTED. LEGAL CHARACTERS ARE NUMBERS,
\*,-,", OR (. IF NONE OF THESE, FATAL ERROR. IF ONE OF THESE,
PROCEED TO SECTION WHICH DECODES CONSTANTS.
IF (ITEMP, GE, INF, AND, ITEMP, LE, INL) GO TO 400
IF (ITEMP, EQ, 1H, OR, ITEST, EQ, 1H, OR, ITEST, EQ, 1H())
GO TO 400
CONTINUE
CALL MESSOT (13)
IEFLAG = 13
RETURN
----------------------------------------------
SUBLOOP TO DECODE VARIABLE NAMES
----------------------------------------------
CONTINUE
STORE THE FIRST CHARACTER
IVNUM = 1
ID8TR (1) = ITEST
----------------------------------------------
SUBLOOP TO GET CHARACTERS OF VARIABLE NAME
----------------------------------------------
GO TO 140
CONTINUE
GET THE NEXT CHARACTER AND CHECK FOR ERRORS
CALL NXTCOL (IF, IEFALG)
IF (IEFLAG, NE, 0) RETURN
SFE IF CHARACTER IS ALPHANUMERIC. IF NOT, ASSUME THAT THE END OF
THE NAME HAS BEEN REACHED.
ITEMP AND (SHIFT (ITEST, = 54), 77B)
IF (ITEMP, GE, IA, AND, ITEMP, LE, IA,)
GO TO 150
GO TO 170
CONTINUE
CHARACTER IS ALPHANUMERIC, SEE IF IT IS NO, 7, IF SO, FATAL ERROR.
IF (IVNUM, GE, 6) GO TO 160
NOT CHARACTER 7. STORE AND GO ON TO THE NEXT.
IVNUM = IVNUM + 1
ID8TR (IVNUM) = ITEST
GO TO 140
160 CONTINUE
CALL MESSOT(3)
IEFLAG=3
RETURN
C
170 CONTINUE
NAME HAS BEEN DECODED, PUT INTO ONE WORD,
CALL MASH(IDSTR,IVNUM,INAME,IEFLAG)
IF(IEFLAG,NE,0) RETURN
C
SCAN THE TABLE TO SEE IF IT IS A LEGAL VARIABLE, IF NOT, 
FATAL ERROR.
DO 180 IM2,INE
180 IF(INAME.EQ.,TABLE(1,1)) GO TO 190
CALL MESSOT(4)
IEFLAG=4
RETURN
C
LEGAL VARIABLE NAME HAS BEEN FOUND, INITIALIZE FOR SUBSEQUENT 
SCAN.
190 CONTINUE
FIRST HERE DENOTES THAT A CONSTANT HAS NOT 
YET BEEN FOUND FOR 
THIS VARIABLE
FIRST=.TRUE.,
IENTRY IS THE SECOND SUBSCRIPT OF THE TABLE ENTRY FOR THIS 
VARIABLE
IENTRY=1
ISBSPT IS THE STORAGE OFFSET FOR THE VARIABLE, IF THE VARIABLE 
IS LOCATED BEGINNING IN MEMORY LOCATION N, THEN THE NEXT 
CONSTANT WILL BE STORED STARTING IN LOCATION N+ISBSPT=1.
ISBSPT=1
NSUBV(I) IS THE VALUE OF THE ITH SUBSCRIPT
DO 200 IM1,4
200 NSUBV(I)=0
NSUB IS THE NUMBER OF SUBSCRIPTS
DIM = .TRUE., IF THE VARIABLE IS DIMENSIONED 
DIM=.FALSE.
DETERMINE ANY SUBSCRIPTING INFORMATION

CHECK THE FIRST CHARACTER AFTER THE VARIABLE NAME. IF IT IS AN EQUALS, NO SUBSCRIPTING INFORMATION, PROCEED TO CONSTANT SCAN.
OTHER LEGAL CHARACTERS ARE BLANK AND LEFT PAREN. IF NEITHER OF THESE, FATAL ERROR.

IF(ITEST, EQ, 1H) GO TO 220
IF(ITEST, EQ, 1H) GO TO 230
IF(ITEST, EQ, 1H) GO TO 400
CALL MESSOT(5)
IFLAG=5
RETURN

CHARACTER WAS BLANK, FETCH NEXT COLUMN AND CONTINUE.

CALL NXTCOL(IF, IFLAG)
IF(IEFLAG, NE, 0) RETURN
GO TO 210

CHARACTER WAS LEFT PAREN, BEGIN SUBSCRIPT DECODING.

CALL NXTCOL(IF, IFLAG)
IF(IEFLAG, NE, 0) RETURN

INITIALIZE, IVSUM IS THE CURRENT VALUE OF THE NUMBER BEING DECODED AND IVSIGN IS ITS SIGN (+1 OR -1).

DIM=TRUE,
NSUB=1
IVNUM=0
IVSIGN=1
DELIM=FALSE,

NO8GN IS THE NO. OF SIGNS ENCOUNTERED IN THE FIELD

+++ SUBLOOP TO DECODE A SUBSCRIPT

GET THE NEXT CHARACTER

CALL NXTCOL(IF, IFLAG)
IF(IEFLAG, NE, 0) RETURN

SEE IF IT IS NUMERIC

ITEM=AND(SHIFT(ITEST, =54), 778)
IF(ITEM, GE, INF, AND, ITEM, LE, INL) GO TO 250
NO, CHECK FOR BLANK,
IF(ITEST, EQ, 'H') GO TO 310
NO, CHECK FOR COMMA,
IF(ITEST, EQ, 'H') GO TO 340
NO, CHECK FOR RIGHT PAREN,
IF(ITEST, EQ, 'H') GO TO 360
CHECK FOR SIGN OF SUBSCRIPT
IF(ITEST, EQ, 'H' OR ITEST, EQ, 'H') GO TO 290
NONE OF THESE, FATAL ERROR,
CALL MESSOT(6)
IEFLAG=6
RETURN
CHARACTER WAS NUMERIC, SEE IF A DELIMITER WAS EXPECTED, IF IT WAS,
FATAL ERROR,
CONTINUE
IF(NOT,DELIM) GO TO 270
CONTINUE
CALL MESSOT(7)
IEFLAG=7
RETURN
SEE IF THE SUBSCRIPT IS TOO LONG, IF MORE THAN 5 DIGITS, FATAL ERROR,
IF(INUM, LT, 5) GO TO 280
CALL MESSOT(43)
IEFLAG=43
RETURN
SUBSCRIPT IS OK SO FAR, UPDATE ITS VALUE AND CONTINUE,
IVNUM=IVNUM+1
IVSUM=IVSUM+10*ITEMP=INF
GO TO 240
A SIGN HAS BEEN FOUND, MAKE SURE THERE IS ONLY ONE,
IF(NOSGN, EQ, 0) GO TO 300
CALL MESSOT(44)
IEFLAG=44
RETURN
300 CONTINUE
C SIGN IS OK, UP THE COUNT, SET THE SIGN VARIABLE, AND CONTINUE SCAN.
310 CONTINUE
C CHARACTER IS A BLANK, IF NO NUMBERS HAVE BEEN FOUND YET, GO ON TO
C THE NEXT CHARACTER.
C IF(IVNUM.EQ.0) GO TO 240
C BLANK CONSIDERED FIELD TERMINATOR, TURN ON THE DELIMITER SWITCH.
C DELIM=.TRUE.,
C 320 CONTINUE
C SUBSCRIPT DECODED, MAKE SURE THERE ARE NO MORE THAN 4,
C IF(NSUB.GT.4) GO TO 330
C SUBSCRIPT IS OK, STORE THE SIGNED VALUE IN THE SUBSCRIPT ARRAY.
C NSUBV(NSUB)=SIGN(IVSUM,IVSIGN)
C REINITIALIZE AND GO SEARCH FOR NEXT SUBSCRIPT
C IVSUM=0
C IVNUM=0
C IVSIGN=1
C NOSGN=0
C NSUB=NSUB+1
C GO TO 240
C MORE THAN FOUR SUBSCRIPTS, FATAL ERROR.
C 330 CONTINUE
C CALL MESSOT(14)
C IEFLAG=14
C RETURN
C 340 CONTINUE
C COMMA ENCOUNTERED, SEE IF A DELIMITER IS EXPECTED.
C IF(DELIM) GO TO 350
C NO, SEE IF A FIELD HAS BEGUN, IF NOT, FATAL ERROR, IF SO, CONSIDER
C FIELD TERMINATED AND GO FINISH PROCESSING FOR THIS SUBSCRIPT.
C IF(IVNUM.NE.0) GO TO 320
C CALL MESSOT(8)
C IEFLAG=8
C RETURN
C DELIMITER EXPECTED AND FOUND, TURN OFF FLAG AND CONTINUE SCAN.
C 350 CONTINUE
DELIM=FALSE.
GO TO 240

RIGHT PAREN FOUND, TERMINATE SUBSCRIPT PROCESSING, CHECK IF PAREN
IS FIELD TERMINATOR.

IF(IVNUM .NE. 0) GO TO 370

NO, CHECK FOR ISOLATED SIGN, IF SO, FATAL ERROR.

IF(NOSGN.EQ.0) GO TO 380
CALL MESSOT(30)
IEFLAG=30
RETURN

370 CONTINUE

YES, STORE THE SUBSCRIPT AND PRETEND LIKE ANOTHER IS TO BE
DECODED.

NSUB(NSUM)*ISIGN(IVSUM,IVSIGN)
NSUB=NSUB+1

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}

380 CONTINUE

LAST SUBSCRIPT COMPLETED, SET NSUB TO THE ACTUAL NO. OF SUBSCRIPTS

NSUB=NSUB-1

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}

390 CONTINUE

SCAN CHARACTERS UNTIL AN "#" IS FOUND, ONLY OTHER VALID CHARACTER
IS A BLANK.

CALL NXTCOL(IF,IEFLAG)
IF(IEFLAG .NE. 0) RETURN
IF(ISTEST.EQ.1H) GO TO 390
IF(ISTEST.EQ.1H=.I) GO TO 400
CALL MESSOT(9)
IEFLAG=#
RETURN

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}

SECTION TO DECODE CONSTANTS

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}

400 CONTINUE

RESET ALL FLAGS

CMPLEX = .TRUE. IF A COMPLEX CONSTANT IS BEING DECODED
CDONE = .TRUE. IF THE SECOND HALF OF A COMPLEX CONSTANT HAS BEEN DECODED BUT THE CLOSING PARENTHESES HAS NOT
IF THE CLOSING PARENTHESIS OF A COMPLEX CONSTANT HAS BEEN FOUND

DB = STORAGE FOR DOUBLE PRECISION CONSTANT
MULT = IF A MULTIPLIER HAS BEEN FOUND
DELIM = IF A DELIMITER IS EXPECTED

COMPLEX = FALSE,
DONE = FALSE,
CPFND = FALSE,
DB = 0,
MULT = FALSE,
DELIM = FALSE

ONE RETURNS TO HERE IF THE FIRST NO. OF A COMPLEX CONSTANT HAS BEEN PROCESSED AND STORED

CONTINUE

HOLLER = IF A HOLLERITH CONSTANT IS BEING PROCESSED
IC = CHARACTER COUNT OF CONSTANT STRING
DOBLE = IF A DOBLE PRECISION CONSTANT IS BEING PROCESSED
FIX = IF A FIXED POINT CONSTANT IS BEING DECODED
ISUM = CURRENT VALUE OF FIXED POINT FIELD
IVNUM = CURRENT NO. OF DIGITS IN THE FIXED POINT FIELD
EXPN = IF AN EXPONENT IS BEING DECODED
XLEFT = VALUE OF FLOATING POINT CONSTANT TO LEFT OF DECIMAL POINT
XRIGHT = VALUE OF FLOATING POINT CONSTANT TO RIGHT OF DECIMAL
NOSGN = NO. OF + OR - SIGNS FOUND IN CONSTANT SO FAR
DONE = IF A DOLLAR SIGN HAS BEEN FOUND
ISIGN = VALUE OF THE LAST SIGN FOUND (+1 OR -1)
PERIOD = IF THE LAST CHARACTER FOUND WAS AN ISOLATED PERIOD

HOLLER = FALSE,
IC = 0,
DOBLE = FALSE,
FIX = TRUE,
ISUM = 0,
IVNUM = 0

IF THIS IS THE FIRST CONSTANT TO BE PROCESSED AFTER AN = SIGN
(FIRST = TRUE) OR A COMPLEX CONSTANT IS BEING DECODED,
ONE MUST FETCH THE NEXT CHARACTER, OTHERWISE, IT HAS PREVIOUSLY BEEN READ.

IF (NOT FIRST .AND. NOT COMPLEX) GO TO 430
CONTINUE

GET THE NEXT CHARACTER AND CHECK FOR ERRORS
CALL NXTCOL(IF,IEFLAG)
IF(IEFLAG,NE,0) RETURN

CONTINUE

DECIDE WHAT TO DO BASED UPON WHAT CHARACTER WAS LAST READ

CHECK FOR A BLANK
IF(ITEST,EQ,1H ) GO TO 470
NO, CHECK FOR A NUMBER,
ITEMP=AND(SHIFT(ITEST,=54),77B)
IF(ITEMP,NE,INF,AND,ITEMP,LE,INL) GO TO 450

NO, CHECK FOR A COMMA,

IF(ITEST,EQ,1H,) GO TO 1050
NO, CHECK FOR A PERIOD,

IF(ITEST,EQ,1H.) GO TO 1140
NO, CHECK FOR A PLUS OR MINUS,

IF(ITEST,EQ,1H+) GO TO 910
IF(ITEST,EQ,1H~) GO TO 990

NO, CHECK FOR A DOLLAR SIGN,
IF(!TEST,EQ,1HS) GO TO 1080

NO, CHECK FOR A T OR F,
IF(ITEST,EQ,1HT) GO TO 1170
IF(ITEST,EQ,1HF) GO TO 1170

NO, CHECK FOR AN E,
IF(ITEST,EQ,1HE) GO TO 1100

NO, CHECK FOR AN ASTERISK,
IF(ITEST,EQ,1HS) GO TO 680

NO, CHECK FOR AN H,
IF(ITEST,EQ,1HH) GO TO 770

NO, CHECK FOR A B,
IF(ITEST,EQ,1HB) GO TO 930

NO, CHECK FOR AN L OR AN R,
IF(ITE8T.EQ.1HL) GO TO 770
IF(ITE8T.EQ.1HR) GO TO 770
C
NO, CHECK FOR A D.
C
IF(ITE8T.EQ.1HD) GO TO 1130
C
NO, CHECK FOR A LEFT PARENTHESES.
C
IF(ITE8T.EQ.1H) GO TO 720
C
NO, CHECK FOR A RIGHT PARENTHESES.
C
IF(ITE8T.EQ.1H)) GO TO 750
C
LEGAL CHARACTER NOT FOUND, FATAL ERROR.
C
CONTINUE
C
CALL MESSOT(12)
C
IEFLAG=12
C
RETURN
C
---
C
CHARACTER WAS A NUMBER
C
---
C
CONTINUE
C
SEE IF A DELIMITER WAS EXPECTED, IF SO, FATAL ERROR.
C
IF(DELIM) GO TO 260
C
RESET ISOLATED PERIOD FLAG
C
PERIOD=FALSE.
C
UPDATE THE VALUE OF THE NUMERIC FIELD
C
IVSUM=IVSUM*10+ITEMP=INF
C
UPDATE THE DIGIT COUNT
C
IVNUM=IVNUM+1
C
UPDATE THE CHARACTER COUNT
C
IC=IC+1
C
SEE IF THIS IS THE THIRTIETH CHARACTER
C
IF(IC.LT.30) GO TO 460
C
YES, FATAL ERROR.
C
GO TO 1160
C
NO, STORE THE CHARACTER AND GO ON TO THE NEXT.
C                     460 CONTINUE
  IDSTR(IC)*ITEST
  GO TO 420
C                     470 CONTINUE
C  IF THE BEGINNING OF A CONSTANT HAS NOT YET BEEN FOUND, GO ON TO
C  THE NEXT CHARACTER
C  IF(FIX.AND. IVNUM.EQ.0) GO TO 420
C  NOT THE BEGINNING OF A CONSTANT, BLANK IS CONSIDERED FIELD
C  TERMINATOR, TURN ON THE FLAG WHICH SAYS STORE THE CONSTANT,
C  SAVE=.TRUE.,
C  SEE IF CURRENT FIELD IS FIXED OR FLOATING
C  IF(.NOT.FIX) GO TO 490
C  FIXED, DO NOT STORE NOW BUT TURN ON DELIMITER FLAG AND CONTINUE
C  CHARACTER SCAN,
C  DELIM=.TRUE.,
C  GO TO 420
C  ENTRY POINT FOR STORING FLOATING POINT NO., IF FIELD IS TERMINATED
C  BY A , OR A $
C                     480 CONTINUE
C  SAVE=.TRUE.,
C                     490 CONTINUE
C  CHECK FOR ISOLATED PERIOD
C  IF(.NOT.PERIOD) GO TO 500
C  YES, FATAL ERROR,
C  CALL MESSOT(29)
C  IEFLAG=29
C  RETURN
C                     500 CONTINUE
C  NO, SEE IF WE ARE PROCESSING AN EXPONENT,
C  IF(.NOT.EXPN) GO TO 530
C  YES, SEE IF IT IS ZERO,
C  IF(IVSUM.EQ.0) GO TO 630
C NO. GIVE IT THE PROPER SIGN.
C IVSUM*SIGN(IIVSUM,IVSIGN)
C SEE IF IT IS WITHIN BOUNDS
C IF(IVSUM.GT.IEMIN.AND.IIVSUM.LT.IEMAX) GO TO 510
C NO, FATAL ERROR.
C CALL MESS0T(26)
C IFLAG=26
C RETURN

510 CONTINUE
C YES, ADJUST THE NO. BASED ON THE EXPONENT AND PROCEED ON
C TO THE GENERAL STORING CODE, USE INTERMEDIATE DOUBLE
C PRECISION IN ALL CASES.
C
C IF(DOUBLE) GO TO 520
C XLEFT*XLEFT*OCONSt**IVSUM
C GO TO 630
C 520 CONTINUE
C DB=DB*OCONSt**IVSUM
C GO TO 630

C CODE TO PROCESS FLOATING POINT NO, WITH NO EXPONENT
C
C 530 CONTINUE
C C CHECK FOR FIELD LONGER THAN 15 CHARACTERS BUT NOT DECLARED
C DOUBLE PRECISION, IF SO, FATAL ERROR.
C C
C IF(IC,GT.NDSP+1.AND.,NOT.DOUBLE) GO TO 600
C C CHECK FOR DOUBLE PRECISION FIELD
C C
C IF(.NOT.DOUBLE) GO TO 590
C C YES, CHECK FOR FIELD LONGER THAN 29 CHARACTERS, IF SO, FATAL ERROR.
C C
C IF(IC,GT.NDPP+1) GO TO 1160
C C ALL OK, INITIALIZE FOR EVALUATION. ICP IS THE NO, OF THE FIELD
C CHARACTER CURRENTLY BEING PROCESSED.
C C
C DB=0.
C ICP=0
C
C 540 ICP=ICP+1
CHECK FOR END OF THE FIELD

IF( ICP .GT. IC ) GO TO 580

NO, PICK UP THE NEXT CHARACTER.

I TEST = IDSTR ( ICP )

SEE IF IT IS A DECIMAL POINT

IF ( I TEST .EQ. "1H" ) GO TO 550

NO, ISOLATE THE CHARACTER LOW ORDER IN THE WORD.

ITEMP = AND ( SHIFT ( I TEST , 54) , 77B )

ADD IT INTO THE CONSTANT

DBP = ITEMP - INF

DB = DB + DCONST + DBP

CONTINUE LOOPING

GO TO 540

A DECIMAL POINT HAS BEEN LOCATED, PREPARE TO EVALUATE THAT PORTION
OF THE CONSTANT TO THE RIGHT OF THE DECIMAL, ICPD IS THE NO.
OF DIGITS TO THE RIGHT OF THE DECIMAL,

DBP = 0.

ICPD = 0

LOOP TO EVALUATE PORTION TO THE RIGHT OF THE DECIMAL

560 ICP = ICP + 1

CHECK FOR THE END OF THE FIELD

IF ( ICP .GT. IC ) GO TO 570

NO, PICK UP THE CHARACTER; ADD IT INTO THE SUM, AND CONTINUE.

I TEST = IDSTR ( ICP )

ITEMP = AND ( SHIFT ( I TEST , 54) , 77B )

DTEMP = ITEMP - INF

DBP = DBP + DCONST + DTEMP

ICPD = ICPD + 1

GO TO 560

END OF THE FIELD ENCOUNTERED, SEE IF THERE WAS ANYTHING NON-ZERO
TO THE RIGHT OF THE DECIMAL.

ITEMP = DB = DZERO

IF ( ITEMP . EQ. 0 ) GO TO 580
YES, SHIFT THE DECIMAL TO THE LEFT AND ADD IT INTO THE PORTION
OF THE ENTIRE CONSTANT TO THE LEFT OF THE DECIMAL.

DBP=DBP+DCONST**(-ICPD)  
DB=DB+DBP

NUMBER IS ENTIRELY DECODED, AFFIX THE PROPER SIGN,

IF(IVSIGN,LT,0) DB=DB

************
GO SEE IF THE NO. IS TO BE STORED
GO TO 630

NO, IS SINGLE PRECISION, SEE IF IT IS ZERO,

IF(IVSUM.EQ.0) GO TO 620

NO, SEE IF IT IS TOO LONG,

IF(IVNUM.LE,NDSP) GO TO 610

YES, FATAL ERROR,

600 CONTINUE

CALL MESSOT(28)
IEFLAG=28
RETURN

610 CONTINUE

NO, STORE IT.

XRIGHT=IVSUM

SHIFT THE DECIMAL POINT OF THE RIGHHAND SIDE TO THE FAR LEFT

XRIGHT=XRIGHT*10.**(IVNUM)

ADD THE LEFT AND RIGHT PARTS

XLEFT=XLEFT*XRIGHT

620 CONTINUE

AFFIX THE SIGN

IF(IVSIGN,LT,0) XLEFT=-XLEFT

IF DOUBLE PRECISION, CONVERT TO THAT FORM

IF(DOUBLE) DB=XLEFT

630 CONTINUE

DECODING FOR BOTH SINGLE AND DOUBLE PRECISION FINISHED,
REINITIALIZE.
IF THE SAVE FLAG IS NOT SET, GO BACK AND EXAMINE THE NEXT CHARACTER.

IF(.,NOT.,SAVE) GO TO 420

SEE IF A COMPLEX CONSTANT IS BEING PROCESSED.

IF(.,NOT.,CMPLEX) GO TO 670

YES, SEE IF THIS IS THE FIRST OR SECOND PART.

IF(CFRST) GO TO 660

SECOND PART, SEE IF THE SECOND PART HAS ALREADY BEEN STORED.

IF(.,NOT.,CDONE) GO TO 650

YES, FATAL ERROR.

640 CONTINUE

CALL MESSOT(42)

IFLAG#42

RETURN

NO, SAVE THE SECOND PART AND SET THE FLAG.

650 CONTINUE

CIMAG#LEFT

CDONE=.TRUE.

GO TO 670

660 CONTINUE

FIRST PART, CHECK FOR NO SECOND PART, IF SO, FATAL ERROR.

IF(CPFND) GO TO 640

SET FLAG TO INDICATE SECOND PART WILL BE PROCESSED NEXT.

CFRST=.FALSE.

SAVE THE FIRST PART.

CREAL#LEFT

IF THE LAST CHARACTER FOUND WAS NOT A COMMA, TURN ON THE DELIMITER=EXPECTED FLAG.

IF(ITEST,NE,1H,) DELIM=.TRUE.

GO BACK AND BEGIN PROCESSING THE SECOND PART.

GO TO 410
C       READY TO STORE CONSTANT, IF IT IS NOT DOUBLE PRECISION OR COMPLEX,  
C       SHIFT IT INTO STORAGE KNOWN TO THE ROUTINE STORE.  
C       IF (.NOT., DOBLE, AND .NOT., CMPLEX) XSUM = XLEFT  
C       STORE THE CONSTANT  
C       CALL STORE (TABLE, IFLAG)  
C       IF ANY ERRORS, QUIT RIGHT HERE  
C       IF (IFLAG .NE. 0) RETURN  
C       IF THE CLOSING $ HAS BEEN FOUND, WE ARE ALL DONE  
C       IF (DONE) RETURN  
C       IF A COMPLEX CONSTANT IS BEING PROCESSED AND THE CLOSING  
C       PARENTHESIS HAS NOT YET BEEN FOUND, CONTINUE ON TO NEXT  
C       CHARACTER  
C       IF ( CMPLEX, AND .NOT., CPFND ) GO TO 420  
C       IF THE LAST CHARACTER FOUND WAS NOT A COMMA, TURN ON THE  
C       DELIMITER FLAG  
C       IF ( ITEST .NE. 1H ) DELIM = TRUE,  
C       GO BACK AND HUNT FOR THE BEGINNING OF ANOTHER VARIABLE NAME  
C       OR CONSTANT  
C       GO TO 90  
C       600 CONTINUE  
C       -------------------------------  
C       CHARACTER IS AN ASTERISK  
C       -------------------------------  
C       690 CONTINUE  
C       MUST HAVE BEEN PRECEDED BY A FIXED POINT NO, IF NOT, FATAL ERROR,  
C       IF (FIX) GO TO 690  
C       CALL MESSOT (17)  
C       IFLAG .EQ. 17  
C       RETURN  
C       690 CONTINUE  
C       SEE IF AN ASTERISK HAS PREVIOUSLY BEEN ENCOUNTERED, IF SO,  
C       FATAL ERROR,  
C       IF (.NOT., MUL) GO TO 700  
C       CALL MESSOT (18)  
C       IFLAG .EQ. 18  
C       RETURN  
C       700 CONTINUE
IF PROCESSING OF A COMPLEX NO. HAS BEGUN, FATAL ERROR
INTERNIDGE(NOT,CMPLEX) GO TO 710
CALL MESSOT(19)
IEFLAG=19
RETURN
710 CONTINUE

ALL CONDITIONS O.K., TURN ON THE FLAG, SAVE THE MULTIPLIER,
REINITIALIZE, AND GO ON TO THE NEXT CHARACTER,
MULT=TRUE,
MULTY=IVSUM
IVNUM=0
IVSUM=0
IVSIGN=1
DELIM=FALSE,
GO TO 420

20 CONTINUE

IF A DELIMITER WAS EXPECTED, FATAL ERROR
IF(DELIM) GO TO 260
SEE IF ANY OTHER FIELD CHARACTERS HAVE BEEN READ, IF SO, FATAL
ERROR,
IF(IVNUM,EQ.0) GO TO 730
CALL MESSOT(20)
IEFLAG=20
RETURN
730 CONTINUE

SEE IF A SIGN HAS BEEN ENCOUNTERED, IF SO, FATAL ERROR,
IF(NOSIGN,EQ.0) GO TO 740
CALL MESSOT(39)
IEFLAG=39
RETURN
740 CONTINUE

EVERYTHING CHECKS, TURN ON THE COMPLEX FLAG, INDICATE THAT THE
FIRST HALF OF THE CONSTANT IS BEING PROCESSED, AND GO ON TO
THE NEXT CHARACTER,
CMPLX=TRUE,
CFRST=TRUE,
GO TO 420

CHARACTER IS A RIGHT PARENTHESIS
IF A RIGHT PARENTHESIS HAS PREVIOUSLY BEEN FOUND, FATAL ERROR

IF(CPFND) GO TO 440

IF STILL PROCESSING A COMPLEX CONSTANT, FATAL ERROR

IF(CFRST, OR, DELIM, OR, FIX) GO TO 640

ALL CHECKS PASSED, INDICATE CLOSING PARENTHESIS FOUND AND GO

STORE THE RESULTS.

CALL MESSOT(34)

RETURN

IF NO CHARACTER COUNT HAS BEEN PREVIOUSLY FOUND, FATAL ERROR
IF(IVNUM.NE.0) GO TO 780
CALL MESSOT(35)
IFLAG=35
RETURN
CONTINUE

780 CONTINUE

C C C IF CHARACTER COUNT IS MORE THAN 10, A MULTIPLIER CANNOT BE USED.
C C C IF 80, FATAL ERROR,
C C IF(.NOT.MULT.OR.IVNUM.LE.10) GO TO 800
CALL MESSOT(36)
IFLAG=36
RETURN

800 CONTINUE

C C C EVERYTHING CHECKS, TURN ON THE HOLLERITH FLAG,
C C HOLLER=TRUE,
C C THE CHARACTER COUNT IS STORED IN IVNUM AND THE TYPE OF FIELD
C C IN ISAVE (H,L,OR R)
C IVNUM=IVSUM
ISAVE=ITEST

C C C SPECIAL PROCESSING IF MORE THAN A WORDS WORTH OF CHARACTERS
C IF(IVNUM.GT.10) GO TO 900
C C C TEN OR LESS CHARACTERS TO PROCESS
C 810 CONTINUE

C C C PICK UP THE REMAINING CHARACTERS AND STORE IN IDSTR
DD 820 #1,IVNUM
CALL NXTCOL(IF,IEFLAG)
IF(IEFLAG.NE.0) RETURN
IDSTR(I)=ITEST

820 CONTINUE

C C C IF TEN CHARACTERS, NO DISTINCTION BETWEEN H,L,AND R
C IF(IVNUM.EQ.10) GO TO 860
C C C CHECK FOR H
C IF(ISAVE.NE.1HH) GO TO 830
IPAD=MH
GO TO 840
830 CONTINUE

C C C NO, CHECK FOR R,
C 840 CONTINUE
C
IF(ISAVE.EQ.1HR) GO TO 870
C
C NO, ASSUME L, PAD END OF WORD WITH ZEROS.
C
IPAD=0
840 CONTINUE
IVP=IVNUM+1
DO 850 I=IVP,10
850 IDSTR(I)=IPAD
C
C UP CHARACTER COUNT TO FULL WORD
C
IVNUM=10
860 CONTINUE
C
C TAKE INDIVIDUAL CHARACTERS AND PUT INTO A SINGLE WORD (IVSUM)
C
CALL MASH(IDSTR, IVNUM, IVSUM, IFLAG)
C
IF(IFLAG.NE.0) RETURN
C
C TURN ON THE DELIMITER FLAG, STORE THE WORD, AND GO ON TO NEXT
C VARIABLE NAME OR CONSTANT
C
C
C TAKE INDIVIDUAL CHARACTERS AND PUT INTO A SINGLE WORD (IVSUM)
C
C
C PUT THE CHARACTERS INTO ONE WORD AND STORE IT
C
GO TO 860
C
C R FIELD, RIGHT JUSTIFY CHARACTERS AND PAD UPPER PART WITH ZEROS.
C
870 CONTINUE
C
C FIELD IS LONGER THAN 10 CHARACTERS
C
GO TO 860
C
C FIELD IS LONGER THAN 10 CHARACTERS
C
900 CONTINUE
C
C CHECK FOR R FIELD, IF SO, FATAL ERROR.
C
IF(ITEST.NE.1HR) GO TO 910
C
910 CONTINUE
DO 920 IM=1,10
CALL NXTCOL(IF,IEFLAG)
IF(IEFLAG,NE,0) RETURN
1500
920 IDSTR(I)=TEST
1502
C
1503 C PUT THEM INTO ONE WORD AND STORE THEM
1504 C
1505 CALL MASH(IDSTR,10,IVSUM,IEFLAG)
1506 IF(IEFLAG,NE,0) RETURN
1507 CALL STORE(TABLE,IEFLAG)
1508 IF(IEFLAG,NE,0) RETURN
1509 C
1510 C REDUCE THE CHARACTER COUNT BY 10 AND SEE IF IT IS NOW LE.10,
1511 C IF SO, GO FINISH PROCESSING THE FIELD. IF NOT, GO BACK AND
1512 C PICK UP THE NEXT 10 CHARACTERS.
1513 C
1514 IVNUM=IVNUM-10
1515 IF(IVNUM.LE,10) GO TO 810
1516 GO TO 910
1517 C
1518 C **************************************************
1519 C CHARACTER IS A B
1520 C **************************************************
1521 C
1522 930 CONTINUE
1523 C
1524 C INSURE THAT PREVIOUS FIELD WAS FIXED POINT, IF NOT, FATAL ERROR.
1525 C
1526 IF(FIX) GO TO 940
1527 CALL MESSOT(21)
1528 IFLAG=21
1529 RETURN
1530 940 CONTINUE
1531 C
1532 C SEE IF PREVIOUS FIELD WAS ZERO, IF SO, NO CONVERSION NECESSARY.
1533 C
1534 IF(IVNUM.EQ,0) GO TO 970
1535 C
1536 C SEE IF PREVIOUS FIELD WAS 20 CHAR OR LESS, IF NOT, FATAL ERROR.
1537 C
1538 IF(IVNUM.LE,20) GO TO 950
1539 CALL MESSOT(40)
1540 IFLAG=40
1541 RETURN
1542 950 CONTINUE
1543 C
1544 C LOOP TO DECODE OCTAL NUMBERS
1545 C **************************************************
1546 C **************************************************
1547 C
1548 C IFSUM WILL BE THE RESULTING NO.
1549 C ISHIFT IS THE NO. OF BITS TO SHIFT THE CURRENT DIGIT TO THE LEFT
1550 C BEFORE OR-ING IT INTO IFSUM
1551 C
DO 960 I=1,IVNUM
C ISOLATE THE CHARACTER LOW ORDER IN TEMPORARY STORAGE
ITEMP=AND(SHIFT(IDSTRI),54),778
C MAKE SURE IT IS NOT AN 8 OR A 9, IF SO,FATAL ERROR.
IF(ITEMP,GT,INF+7) GO TO 980
C CONVERT,SHIFT,AND OR
ITEMP=ISHIFT*3
ITEMP=SHIFT(ITEMP-INF,ISHFT)
IVSUM=OR(IVSUM,ITEMP)
960 CONTINUE
970 CONTINUE
C +++++++++++++++++++++++++++++++++++++++++++++++++++++++
C TURN ON THE DELIMITER FLAG AND GO STORE THE CONVERTED CONSTANT
DELIM=TRUE.
GO TO 1070
980 CONTINUE
C ILLEGAL CHARACTER IN OCTAL CONSTANT
CALL MESSOT(41)
IENEXT=41
RETURN
C -----------------------------
C CHARACTER IS + OR =
C -----------------------------
990 CONTINUE
C IF A DELIMITER WAS EXPECTED,FATAL ERROR
IF(DELIM) GO TO 260
C IF A SIGN HAS ALREADY BEEN FOUND,FATAL ERROR
IF(NOSGN,NE,9) GO TO 1000
C SEE IF FIXED POINT FLAG IS STILL SET
IF(.NOT,FIX) GO TO 1020
C YES, SIGN BETTER BE FIRST CHARACTER IN THE FIELD OR FATAL ERROR.
IF(IVNUM,EQ,0) GO TO 1010
1000 CONTINUE
CALL MESSOT(22)
1609       `IEFLAG=22
1610     RETURN
1611 1010 CONTINUE
1612       C
1613       C SET IVSIGN BASED ON + OR =, PLUS IS THE DEFAULT.
1614       C
1615      IF(ITEST,EQ,14=) IVSIGN=1
1616      C
1617      INDICATE A SIGN HAS BEEN FOUND AND GO LOOK AT THE NEXT
1618      C CHARACTER
1619      C
1620      NOIGN=1
1621     GO TO 420
1622 1020 CONTINUE
1623       C
1624       C SIGN FOUND BUT FLOATING POINT NO, BEING PROCESSED, EXPONENT FLAG
1625       C BETTER BE SET OR FATAL ERROR.
1626       C
1627      IF(FXPN) GO TO 1040
1628 1030 CONTINUE
1629     CALL MESSOT(27)
1630 `IEFLAG=27
1631     RETURN
1632 1040 CONTINUE
1633       C
1634       C EXPONENT FLAG SET, SEE IF ANY CHARACTERS HAVE BEEN ENCOUNTERED
1635       C AFTER THE E OR D, IF SO, FATAL ERROR.
1636       C
1637      IF(IVNUM.NE,0) GO TO 1050
1638       C
1639      ALL O.K., GO SET THE SIGN AND CONTINUE.
1640       C
1641 1050 CONTINUE
1642       C
1643       C -------------------------------
1644       C CHARACTER IS A COMMA
1645       C -------------------------------
1646 1060 CONTINUE
1647       C
1648       C SEE IF THE DELIMITER-EXPECTED FLAG IS SET
1649       C
1650      IF(DELIM) GO TO 1090
1651       C
1652       C NO, SEE IF THE FIXED POINT FLAG IS SET, IF NOT, COMMA SIGNALS
1653       C TERMINATION OF FLOATING POINT FIELD, GO FINISH DECODING THE
1654       C NO, AND STORE IT.
1655       C
1656      IF(.NOT.FIX) GO TO 480
1657 1080 CONTINUE
1658       C
1659       C FIXED POINT FLAG IS SET, CHECK FOR MORE THAN 14 CHARACTERS IN THE
1660       C FIELD, IF SO, FATAL ERROR.
1661       C
1662      IF(IVNUM,GT,NDSP) GO TO 600
1663       C
1664      C CHECK FOR NO CHARACTERS FOUND, IF SO, FATAL ERROR.
C IF(IVNUM,NE,0) GO TO 1070
1668 CALL MESSOT(30)
1669 IEFLAG=30
1670 RETURN
1671 CONTINUE
1672 C
1673 C SEE IF COMPLEX FLAG IS SET, IF SO, INTEGER FIELD IN COMPLEX
1674 C CONSTANT = FATAL ERROR.
1675 C
1676 C IF(CMPEX) GO TO 660
1677 C
1678 C ALL IS O.K., AFFIX THE SIGN AND STORE THE CONSTANT.
1679 C
1680 C IF(IVSIGN,LT,0) IVSUM=-IVSUM
1681 C CALL STORE(TABLE,IEFLAG)
1682 C IF(IEFLAG,NE,w) RETURN
1683 C
1684 C IF DOLLAR SIGN WAS LAST CHARACTER ENCOUNTERED, ALL DONE, IF NOT,
1685 C GO ON AND LOOK FOR BEGINNING OF NEXT CONSTANT OR VARIABLE
1686 C NAME.
1687 C
1688 C IF(DONE) RETURN
1689 C
1690 C -------------------------------
1691 C CHARACTER IS "$"
1692 C -------------------------------
1693 C
1694 C 1080 CONTINUE
1695 C
1696 C SET THE DONE FLAG, IF FLOATING POINT NO. IS BEING PROCESSED,
1697 C VIEW "$" AS FIELD TERMINATOR AND GO FINISH PROCESSING THE NO.
1698 C IF FIXED POINT, GO PERFORM CHECKS AND STORE THE CONSTANT.
1699 C
1700 C DONE=TRUE.
1701 C IF(,NOT,FIX) GO TO 480
1702 C GO TO 1060
1703 C
1704 C JUMP TO HERE IF DELIMITER EXPECTED AND FOUND
1705 C
1706 C
1707 C 1090 CONTINUE
1708 C
1709 C RESET THE DELIMITER=EXPECTED FLAG
1710 C
1711 C DELIM=.FALSE.
1712 C
1713 C SEE IF COMPLEX CONSTANT IS BEING PROCESSED, IF NOT, ONE CAN ONLY
1714 C GET TO HERE WHILE PROCESSING A FIXED POINT NO, GO STORE IT.
1715 C
1716 C IF(,NOT,CMPEX) GO TO 1060
1717 C
1718 C COMPLEX CONSTANT BEING PROCESSED, IF CHARACTERS FOUND BEFORE
1719 C DELIMITER, FATAL ERROR.
1723   IF(IVNUM,NE,0) GO TO 640
1724   C    EVERYTHING OK, GO ON AND LOOK AT NEXT CHARACTER.
1725   C    GO TO 420
1726   C    -------------------------------------------------------------------
1727   C    CHARACTER IS AN E
1728   C    -------------------------------------------------------------------
1729   C    1100 CONTINUE
1730   C    -------------------------------------------------------------------
1731   C    IF A DELIMITER WAS EXPECTED, FATAL ERROR
1732   C    IF(DELIM) GO TO 260
1733   C    -------------------------------------------------------------------
1734   C    SEE IF MORE THAN 19 CHARACTERS HAVE BEEN FOUND IN THE PRECEDING
1735   C    FIELD. IF SO, E SHOULD HAVE BEEN D, FATAL ERROR.
1736   C    IF(IC,GT,NDSP+1) GO TO 600
1737   C    -------------------------------------------------------------------
1738   C    IF(IVC,GT,NDSP+1) GO TO 600
1739   C    1110 CONTINUE
1740   C    -------------------------------------------------------------------
1741   C    ALL OK, TURN ON THE EXPONENT FLAG AND RESET THE SIGN COUNT.
1742   C    EXPNPTRUE.
1743   C    NDSEXN0
1744   C    -------------------------------------------------------------------
1745   C    SEE IF THE FIELD CURRENTLY BEING PROCESSED IS FIXED POINT
1746   C    IF(,NOT,FIX) GO TO 1120
1747   C    -------------------------------------------------------------------
1748   C    YES, SEE IF ANY CHARACTERS HAVE PREVIOUSLY BEEN FOUND.
1749   C    IF(IVNUM,NE,0) GO TO 1150
1750   C    NO, DEFAULT TO 1. (AND USE A SIGN IF ONE WAS FOUND)
1751   C    IVNUM=1
1752   C    IVSUM=1
1753   C    IF(IVSIGN,LT,0) IVSUM=IVSUM
1754   C    -------------------------------------------------------------------
1755   C    RESET THE SIGN
1756   C    IVSIGN=1
1757   C    -------------------------------------------------------------------
1758   C    GO FINISH PROCESSING PART TO LEFT OF THE EXPONENT
1759   C    GO TO 1150
1760   C    1120 CONTINUE
1761   C    -------------------------------------------------------------------
1762   C    FLOATING POINT FIELD. GO FINISH PROCESSING PART TO THE LEFT OF
1763   C    THE EXPONENT.
1764   C    -------------------------------------------------------------------
1765   C    SAVEF,FALSE.
1766   C    GO TO 530
1767   C    -------------------------------------------------------------------
1768   C    -------------------------------------------------------------------
1769   C    -------------------------------------------------------------------
CHARACTER IS A D

CONTINUE

IF A DELIMITER WAS EXPECTED, FATAL ERROR

IF (DELIM) GO TO 260

SEE IF COMPLEX CONSTANT IS BEING PROCESSED, IF SO, FATAL ERROR.

IF (COMPLEX) GO TO 640

EVERYTHING CHECKS, SET THE DOUBLE PRECISION FLAG AND GO FINISH

INITIALIZING FOR THE EXPONENT.

DOBLE=.TRUE.,

GO TO 1110

CONTINUE

CHARACTER IS A PERIOD

CONTINUE

SEE IF DELIMITER WAS EXPECTED, IF SO, FATAL ERROR.

IF (DELIM) GO TO 260

IF FLOATING POINT FLAG IS ALREADY SET, FATAL ERROR

IF (FIX) GO TO 1150

CALL MESSOT (25)

IEFLAG=25

RETURN

CONTINUE

CHARACTER IS A PERIOD

CONTINUE

SEE IF DELIMITER WAS EXPECTED, IF SO, FATAL ERROR.

IF (DELIM) GO TO 260

IF FLOATING POINT FLAG IS ALREADY SET, FATAL ERROR

IF (FIX) GO TO 1150

CALL MESSOT (25)

IEFLAG=25

RETURN

CONTINUE

IF MORE THAN 28 CHARACTERS PREVIOUSLY FOUND, FATAL ERROR

IF (IVNUM .GT. NDDP) GO TO 1160

IF THE PERIOD IS THE FIRST CHARACTER IN THE FIELD, SET THE

ISOLATED PERIOD FLAG

IF (IVNUM .EQ. 0) PERIOD = .TRUE.,

NO PROBLEMS, STORE THE PART TO THE LEFT OF THE PERIOD AS A

FLOATING POINT NO,

XLEFT=IVSUM

IVSUM=0

RESET THE FIELD FLAGS

IVNUM=0

TURN ON THE FLOATING POINT FLAG
C  FIX=.FALSE.,
C  STORE THE CHARACTER FOR LATER DOUBLE PRECISION PROCESSING IF
C  NECESSARY
C  IC=IC+1
C  IDSTR(IC)=IH.
C  IF THE DOUBLE PRECISION FLAG IS ON, STORE PART TO LEFT OF DECIMAL
C  IN DOUBLE PRECISION FORM
C  IF(DOUBLE) DB=LEFT
C  GO ON AND LOOK AT THE NEXT CHARACTER
C  GO TO 420
C  FATAL ERROR, TOO MANY DIGITS IN CONSTANT.
C  1160 CONTINUE
C  CALL MESSOT(38)
C  IFLAG=38
C  RETURN
C  CHARACTER IS T OR F
C  -------------------------------
C  1170 CONTINUE
C  IF DELIMITER IS EXPECTED, FATAL ERROR
C  IF(DELIM) GO TO 260
C  A PERIOD SHOULD HAVE PRECEDED EITHER T OR F, IF NOT, FATAL ERROR,
C  IF(PERIOD) GO TO 1180
C  CALL MESSOT(31)
C  IFLAG=31
C  RETURN
C  1180 CONTINUE
C  SAVE THE CHARACTER AND GO GET THE NEXT ONE
C  ISAVE=ITEST
C  CALL NXTCOL(IF,IFLAG)
C  IF(IFLAG.NE.0) RETURN
C  SEE IF IT WAS A PERIOD
C  IF(ITEST.NE.1H) GO TO 1210
C  YES, SHORTENED FORM OF LOGICAL CONSTANT BEING USED
C  1190 CONTINUE
c set the delimiter-expected flag
1895 c delim=true,
1896 c pick up either a true or false constant
1897 c
1898 c ivsum=lcont
1899 c if(isave,eq,1hf) ivsum=lcont
1900 c if (ivsign,ge,0) go to 1200
1901 c if(ivsum,eq,lconf) ivtemp=lcont
1902 c if(ivsum,eq,lcont) ivtemp=lconf
1903 c ivsum=ivtemp
1904 c
1905 c store the constant and proceed on to next variable or constant
1906 c
1907 c call store(table,ieflag)
1908 c if(ieflag,ne,0) return
1909 c go to 90
1910 c
1911 c long form of logical constant being used, check for beginning
1912 c of true or false constant
1913 c
1914 c if(isave,eq,lhf) go to 1250
1915 c true, next character better be r or fatal error.
1916 c
1917 c if (itest,eq,1hr) go to 1230
1918 c 1220 continue
1919 c call mesfot(l0)
1920 c ieflag=10
1921 c return
1922 c 1230 continue
1923 c tr found, get next character and check for u, if not, fatal error.
1924 c
1925 c call nxtcoll(if,ieflag)
1926 c if(ieflag,ne,0) return
1927 c if (itest,ne,1lu) go to 1220
1928 c 1240 continue
1929 c
1930 c all but e, successfully found, check for these. if not found,
1931 c fatal error. if found, go set up and store constant.
1932 c
1933 c call nxtcoll(if,ieflag)
1934 c if (ieflag,ne,0) return
1935 c if (itest,ne,1he) go to 1220
1936 c call nxtcoll(if,ieflag)
1937 c if (ieflag,ne,0) return
1938 c if (itest,ne,1he) go to 1220
1939 c go to 1190
1940 c 1250 continue
1941 c
SUBROUTINE MESSOT(IEFLAG)

ROUTINE TO DO PRINTING FOR NAHLST

IEFLAG IS THE ERROR FLAG (SEE NAHLST)

WRITTEN BY J.L. NORTON, LASL T=3, 1974

COMMON/ARRCON/ICHAR(80), ISUB, ITEST, NSUB, NSUBV(4), IENTRY, ISSBPT,
CREAL, CIMAG, CIMPLEX, DIM, MULTSV, IVSUM, MULT, LIST, HOLLER, DOBLE
COMMON/IOTAB/NF, IFTAB
COMMON/IOJTAB/NFE, IFTAB
 INTEGER GETIT, SHIFT, OR, AND, COMP
 DIMENSION ILINE(8)
 DATA IPT/IR/

LOOP OVER ALL OUTPUT FILES

IF(IEFLAG,NE,1) GO TO 2
NFP=NF
IFT=IFTAB
GO TO 3
CONTINUE
NFP=NFE
IFT=IFTAB
CONTINUE
IF(NFP, EQ, 9) NFP=1
DO 500 I=1, NFP
IF NO. OF FILES IS 0, GO DEFAULT TO OUT
IF(IFT, NE, 0) GO TO 10
IPX=0
GO TO 2O
CONTINUE
C      PICK UP THE NEXT FILE NAME
37 C  ITST=IFT+I=1
38 C  IPX=GETIT(ITST)
39 C  20 CONTINUE
40 C  PRINT A ROW OF ASTERISKS UNLESS JUST PRINTING THE CARD
41 C  IF(IEFLAG,GT,1) GO TO 30
42 C  WRITE(IPX,970)
43 C  30 CONTINUE
44 C  PRINT THE CARD ITSELF
45 C  WRITE(IPX,510) ICHAR
46 C  ALL DONE IF JUST PRINTING THE CARD
47 C  IF(IEFLAG,GT,1) GO TO 500
48 C  ***********************
49 C  SET UP POINTER TO AREA OF CARD CAUSING ERROR
50 C  ***********************
51 C  BLANK OUT THE POINTER LINE
52 C  DO 40 J=1,8
53 C  N LINE(J)=1H
54 C  APPLY 80 AS UPPER BOUND ON CARD COLUMN NO.
55 C  ISUBX=ISUB
56 C  IF(ISUBX,GT,80) ISUBX=80
57 C  DETERMINE WORD COUNT (L) AND CHARACTER COUNT WITHIN WORD (K)
58 C  OF CARD WORD CONTAINING THE ERROR
59 C  K=MOD(ISUBX,10)
60 C  L=ISUBX/10
61 C  IF(K,NE,0) L=L+1
62 C  IF(K,GT,0) K=10
63 C  SHIFT THE POINTER TO THE PROPER COLUMN POSITION
64 C  KX=10*K
65 C  IC=SHIFT(IPT,IKX)
66 C  MASK OUT THE PROPER POSITION IN THE LINE AND INSERT THE POINTER
67 C  IM=SHIFT(77B,IKK)
68 C  ILINE(L)=OR(IC,AND(ILINE(L),COMP(IM)))
69 C  PRINT THE POINTER LINE
70 C  WRITE(IPX,520) ILINE
GO PRINT THE PROPER ERROR MESSAGE

GO TO (500,50,60,70,80,90,100,110,120,130,140,150,160,170,180,190,200,210,220,230,240,250,260,270,280,290,300,310,320,330,340,350,360,370,380,390,400,410,420,430,440,450,460,470,480), IFLAG

50 WRITE(IPX,530)
100 GO TO 490
101 60 WRITE(IPX,540)
102 GO TO 490
103 70 WRITE(IPX,550)
104 GO TO 490
105 80 WRITE(IPX,560)
106 GO TO 490
107 90 WRITE(IPX,570)
108 GO TO 490
109 100 WRITE(IPX,580)
110 GO TO 490
111 110 WRITE(IPX,590)
112 GO TO 490
113 120 WRITE(IPX,600)
114 GO TO 490
115 130 WRITE(IPX,610)
116 GO TO 490
117 140 WRITE(IPX,620)
118 GO TO 490
119 150 WRITE(IPX,630)
120 GO TO 490
121 160 WRITE(IPX,640)
122 GO TO 490
123 170 WRITE(IPX,650)
124 GO TO 490
125 180 WRITE(IPX,660)
126 GO TO 490
127 190 WRITE(IPX,670)
128 GO TO 490
129 200 WRITE(IPX,680)
130 GO TO 490
131 210 WRITE(IPX,690)
132 GO TO 490
133 220 WRITE(IPX,700)
134 GO TO 490
135 230 WRITE(IPX,710)
136 GO TO 490
137 240 WRITE(IPX,720)
138 GO TO 490
139 250 WRITE(IPX,730)
140 GO TO 490
141 260 WRITE(IPX,740)
142 GO TO 490
143 270 WRITE(IPX,750)
144 GO TO 490
145 280 WRITE(IPX,760)
146 GO TO 490
147 290 WRITE(IPX,770)
148 GO TO 490
149 300 WRITE(IPX,780)
150 GO TO 490
151 WRITE(IPX,790)
152 GO TO 490
153 WRITE(IPX,800)
154 GO TO 490
155 WRITE(IPX,810)
156 GO TO 490
157 WRITE(IPX,820)
158 GO TO 490
159 WRITE(IPX,830)
160 GO TO 490
161 WRITE(IPX,840)
162 GO TO 490
163 WRITE(IPX,850)
164 GO TO 490
165 WRITE(IPX,860)
166 GO TO 490
167 WRITE(IPX,870)
168 GO TO 490
169 WRITE(IPX,880)
170 GO TO 490
171 WRITE(IPX,890)
172 GO TO 490
173 WRITE(IPX,900)
174 GO TO 490
175 WRITE(IPX,910)
176 GO TO 490
177 WRITE(IPX,920)
178 GO TO 490
179 WRITE(IPX,930)
180 GO TO 490
181 WRITE(IPX,940)
182 GO TO 490
183 WRITE(IPX,950)
184 GO TO 490
185 WRITE(IPX,960)
186 490 CONTINUE

C PRINT OTHER ROW OF ASTERISKS
187 WRITE(IPX,970)
188 500 CONTINUE
189 RETURN

C
190 510 FORMAT(1H,80A1)
191 520 FORMAT(1H,8A10)
192 530 FORMAT(1H,41HMORE THAN SIX CHARACTERS IN NAMELIST NAME)
193 540 FORMAT(1H,45HMORE THAN SIX CHARACTERS IN NAMELIST VARIABLE)
194 550 FORMAT(1H,39HNAMELIST VARIABLE NOT IN NAMELIST TABLE)
195 560 FORMAT(1H,50HILLEGAL CHARACTER FOLLOWING NAMELIST VARIABLE NAME)
196 570 FORMAT(1H,30HILLEGAL CHARACTER IN SUBSCRIPT)
197 580 FORMAT(1H,20HDELIMITER EXPECTED)
198 590 FORMAT(1H,25HUNEXPECTED DELIMITER)
199 600 FORMAT(1H,20HILLEGAL LOGICAL VARIABLE)
200 610 FORMAT(1H,45HNAMELIST NAME NOT FOUND IMMEDIATELY AFTER $)
201 620 FORMAT(1H,34HILLEGAL CHARACTER IN THIS POSITION)
202 630 FORMAT(1H,44HFIRST CHARACTER OF NAMELIST VARIABLE ILLEGAL)
SUBROUTINE STORE_TABLE(IEFLAG)

ROUTINE TO STORE THE CONSTANT LAST DECODED

WRITTEN BY J.L. NORTON, LASL T-5, 1974

COMMON/ARRCON/ICCHAR(80),ISUB, ITEST,NSUB,NSUBV(4), ENTRY, I88PT,
1CREAL, CIMAG, CPLEX, DIM, MULTSV,IVSUM,MULT,LIST,HOLLER, DOBLE
9
LOGICAL MULT, CPLEX, DIM, DOBLE
DIMENSION TABLE(3,1)
11
INTEGER TABLE, SHIFT, AND, OR, COMP
12
DATA CREAL/80/
13

GET THE VARIABLE SUBSCRIPT INFORMATION
IAREN=TABLE(3, Entry)
GET THE ABSOLUTE ADDRESS OF THE VARIABLE
IARADD=TABLE(2, Entry)
ISBSPT IS THE MEMORY OFFSET (+1) FOR THE VARIABLE BEING READ, IT
IS 1 UPON THE ENTRY TO STORE FOR A GIVEN VARIABLE NAME, SOME
CHECKING IS DONE ONLY UPON THE FIRST ENTRY, SEE IF THIS IS SO,
IF(ISBSPT.LE.1) GO TO 10

ISBSPT IS THE MEMORY OFFSET (+1) FOR THE VARIABLE BEING READ, IT
IS 1 UPON THE ENTRY TO STORE FOR A GIVEN VARIABLE NAME, SOME
CHECKING IS DONE ONLY UPON THE FIRST ENTRY, SEE IF THIS IS SO,
IF(ISBSPT.LE.1) GO TO 10

ISBSPT IS THE MEMORY OFFSET (+1) FOR THE VARIABLE BEING READ, IT
IS 1 UPON THE ENTRY TO STORE FOR A GIVEN VARIABLE NAME, SOME
CHECKING IS DONE ONLY UPON THE FIRST ENTRY, SEE IF THIS IS SO,
IF(ISBSPT.LE.1) GO TO 10

IF(IAREN.EQ.0) GO TO 60
EVERYTHING CHECKS, CONTINUE ON WITH THE STORE.
GO TO 20
10 CONTINUE

FIRST ENTRY, SEE IF A MULTIPLIER IS IN EFFECT OR VARIABLE HAS
SUBSCRIPTS.
IF(MULT.OR.DIM) GO TO 50
NEITHER CONDITION HOLDS, CONTINUE ON WITH THE STORE.
20 CONTINUE

SET THE MULTIPLIER COUNT, IF NO MULTIPLIER, USE 1.
IF(.NOT.MULT) MULTSV=1
LOOP FOR MULTIPLE STORING
DO 40 I=1,MULTSV

GET THE ACTUAL ABSOLUTE ADDRESS OF THE LOCATION INTO WHICH THE
STORE IS TO OCCUR
IADD=IARADD+ISBSPT

SPECIAL HANDLING FOR DOUBLE OR COMPLEX
IF(COMPLEX.OR.DOUBLE) GO TO 30
ORDINARY ONE ELEMENT VARIABLE, CARRY OUT THE STORE.
CALL STORIT(IVSUMP,IADD)
INCREMENT THE POINTER AND GO TO THE END OF THE LOOP
ISBSPT=ISBSPT+1
GO TO 40
30 CONTINUE
C EITHER DOUBLE OR COMPLEX, STORE TWO WORDS,
C CALL STORIT(CREAL,IADD)
C CALL STORIT(CIMAG,IADD+1)
ISBSPT=ISBSPT+2
40 CONTINUE
C ALL DONE
50 CONTINUE
C VARIABLE HAS SUBSCRIPTS OR CONSTANT HAS A MULTIPLIER, VARIABLE
C MUST BE DIMENSIONED, IF NOT, FATAL ERROR,
C IF(IAREN7,NE,0) GO TO 70
60 CONTINUE
60 CALL MESSOT(15)
IEFLAG=15
RETURN
C IF VARIABLE IS NOT SUBSCRIPTED, ALL IS O.K, CARRY OUT THE STORE,
C IF(.NOT.,DIM) GO TO 20
100 C VARIABLE IS SUBSCRIPTED, GET THE TABLE DIMENSION INFORMATION,
101 C ISUB1=AND(IAREN7,777777B)
102 C ISUB2=AND(SHIFT(IAREN7,-18),777777B)
103 C ISUB3=AND(SHIFT(IAREN7,-36),777777B)
104 C CHECK FOR SUBSCRIPTING ERRORS, IS THERE A FOURTH SUBSCRIPT BUT
C NO THIRD DIMENSION,
105 C IF(NSUB,NE,4,AND,ISUB3,NE,0) GO TO 80
106 C IS THERE A THIRD SUBSCRIPT BUT NO SECOND DIMENSION
107 C IF(NSUB,NE,3,AND,ISUB2,NE,0) GO TO 80
108 C IS THERE A SECOND SUBSCRIPT BUT EITHER NO FIRST DIMENSION OR A
C FIRST DIMENSION INDICATING A SINGLY-SUBSCRIPTED ARRAY
109 C IF(NSUB,NE,2,AND,(ISUB1,NE,1,OR,ISUB1,NE,0)) GO TO 80
110 C IS THERE A SINGLE SUBSCRIPT BUT NO SUBSCRIPTING INFORMATION GIVEN
111 C IF(NSUB,NE,1,AND,ISUB1,NE,0) GO TO 80
112 C IF N SUBSCRIPTS ARE GIVEN, DIMENSIONAL INFORMATION IS USED FOR
C THE FIRST N=1
113 C IF(NSUBV(4),NE,0) ISUB3=0
C
SUBROUTINE MASH(IDSTR, IVNUM, INAME, IEFLAG)

C ROUTINE TO CONVERT AN ARRAY IDSTR CONTAINING ONE SIX-BIT
CHARACTER PER WORD, LEFT-JUSTIFIED, INTO ONE WORD, INAME,
CONTAINING THE SAME CHARACTERS LEFT TO RIGHT WITH BLANK FILL
AT THE RIGHT END IF NEEDED. IVNUM CHARACTERS ARE PROCESSED
WHERE IVNUM MUST BE LE10,

WRITTEN BY J. L. NORTON, LASL T=3, 1974

DIMENSION IDSTR(1)
INTEGER SHIFT, AND, OR, COMP
DATA MASK/7700000000000000/0

C CHECK THE CHARACTER COUNT, IF GT, 10, FATAL ERROR.

C IF(IVNUM.LE.10) GO TO 10
CALL MESSOT(45)
IEFLAG=#5
RETURN
CONTINUE

C INITIALIZE THE PACKED WORD
INAME=1H

C LOOP OVER ALL CHARACTERS

129 IF(NSUBV(3).EQ.0) ISUB2=0
130 IF(NSUBV(2).EQ.0) ISUB1=0
131 C CALCULATE THE OFFSET FROM THE SUBSCRIPTING INFORMATION
132 C
133 C ISBSPT=NSUBV(1)+ISUB1*(NSUBV(2)-1)+ISUB2*(NSUBV(3)-1)+ISUB3*(NSUBV(4-1)))
135 C
136 C MODIFY THE OFFSET FOR COMPLEX VARIABLES
137 C
138 C IF(COMPLEX) ISBSPT=2*ISBSPT=1
139 C
140 C GO TO BACK AND DO THE STORE
141 C
142 C
143 C GO TO 20
144 C
145 C FATAL SUBSCRIPTING ERROR
146 C
147 C CALL MESSOT(16)
148 C
149 C IEFLAG=16
150 C
151 C

END
28 C DO 20 I=1,IVNUM
29 C
30 C SHIFT MASK INTO PROPER POSITION
31 C ITEMP=SHIFT(77B,60-6*I)
32 C
33 C SHIFT CHARACTER INTO PROPER POSITION
34 C ITEMP=SHIFT(IDSTR(I)+6*(I=1))
35 C
36 C DROP CHARACTER INTO INAME, FIRST MASK OUT THE BLANK, THEN ISOLATE
37 C THE CHARACTER AND DROP IT INTO THE PREPARED SLOT.
38 C
39 C 20 INAME=OR(AND(INAME,COMP(ITEMP)),AND(ITEMP,ITEMP))
40 C
41 C ALL DONE
42 C RETURN
43 C END

SUBROUTINE NAMPRT(NFX,FTABX)
C
C ROUTINE TO SET THE FILES ON WHICH THE NAMLST ROUTINE WILL WRITE
C WHENEVER ITS DOES OUTPUT OTHER THAN ERROR MESSAGES (PS OR CS).
C THE CALLING SEQUENCE IS CALL NAMPRT(N,TABLE) WHERE
C N = NO. OF TABLE ENTRIES
C TABLE = ARRAY OF FILE NAMES, EITHER IN THE FORM OF
C LEFT-JUSTIFIED, ZERO-FILLED HOLLERITH
C CONSTANTS OR INTEGERS.
C
C FOR EXAMPLE, IF ONE WANTED TO WRITE BOTH ON FILM AND PAPER,
C THE TABLE WOULD APPEAR AS
C DATA TABLE/4LFILM,3LOUT/.
C IF FILM WAS EQUIVALENCED TO FSET12 AND OUT TO FSET6,
C AN ALTERNATE FORM WOULD BE
C DATA TABLE/12,6/
C AND THE CALL WOULD BE
C CALL NAMPRT(2,TABLE).
C
C IF NAMPRT IS NOT CALLED BEFORE, NAMLST, THE LATTER WILL
20 C DEFAULT ALL OUTPUT TO OUT.
C
21 C WRITTEN BY J. L. NORTON, LASL T=3, 1974
C
22 COMMON/IOTAB/NF,IFLOC
23 INTEGER FTABX(I)
24 DATA NF, IFLOC/0,0/
25 IF(NFX,LE,6) RETURN
26 NF=NFX
27 IFLOC=LOCF(FTABX)
SUBROUTINE ERRPRT(NFX, FTABX)

ROUTINE ANALOGOUS TO NAMPR7 EXCEPT FOR PRINTING ERROR MESSAGES

WRITTEN BY J. L. NORTON, LASL T=3, 1975

COMMON/JOTAB/NF, ILOC
INTEGER FTABX(1)
DATA NF, ILOC/0, 0/
IF(NFX, LE, 0) RETURN
NF=NF
ILOC=LOC(FTABX)
RETURN
END

SUBROUTINE TABDEF(TABLE, NAME, ITSIZE, IERRT)

ROUTINE TO INITIALIZE A NAMELIST TABLE

TABLE = ARRAY WHICH WILL BE USED TO STORE NAMELIST DATA NAMES
AND RELATED INFORMATION; IT MUST BE 3*(N+1) ELEMENTS
LONG WHERE N IS THE NUMBER OF DISTINCT NAMELIST
VARIABLES TO BE READ UNDER THIS NAME.
NAME = NAMELIST NAME, LEFT=JUSTIFIED, BLANK=FILLED HOLLERITH, IF
ONE WERE GOING TO READ CARDS OF THE FORM CODEXN,
NAME=CODEIN,
ITSIZE = SECOND SUBSCRIPT OF THE TABLE ARRAY
(MAXIMUM NO. OF TABLE ENTRIES+1)
IERRT = VARIABLE SET BY ASSIGNED GO TO AS RETURN POINT IF TABLE
INITIALIZATION ENCOUNTERS AN ERROR CONDITION

WRITTEN BY J. L. NORTON, LASL T=3, 1974

INTEGER TABLE
DIMENSION TABLE(3,1)
COMMON/ERRORC/IERRTP
TABLE(1,1)=NAME
TABLE(2,1)=ITSIZE
TABLE(3,1)=0
IERRTP=IERRT
RETURN
END
SUBROUTINE TABSET(TABLE, NAME, LOCA, IEFLAG, NSUB, NSUB1, NSUB2, NSUB3)
C ROUTINE TO ADD A VARIABLE TO A NAMELIST TABLE
C TABLE = AN ARRAY WHICH HAS BEEN INITIALIZED BY A CALL TO TABDEF
C NAME = NAME OF VARIABLE TO BE ADDED, AS A HOLLERITH CONSTANT
C LOCA = THE VARIABLE ITSELF
C IEFLAG = ERROR INDICATOR
C = RETURNED ZERO IF NO ERROR WAS FOUND
C = RETURNED WITH THE CONTENTS OF THE VARIABLE =NAME=
C LEFT-JUSTIFIED (LEFT 6 CHARACTERS) AND THE ERROR NO.
C RIGHT-JUSTIFIED IF AN ERROR OCCURRED
C NSUB = NUMBER OF SUBSCRIPTS (0, 1, 2, 3, OR 4)
C NSUB1 = FIRST ARRAY DIMENSION
C NSUB2 = SECOND ARRAY DIMENSION
C NSUB3 = THIRD ARRAY DIMENSION
C = (IF THE NAMELIST VARIABLE HAS N SUBSCRIPTS (N,LE,4),
C THEN NSUB1, NSUB2,..., NSUBN=1 ARE NEEDED)
C THE ACTUAL FORM OF THE TABLE IS AS FOLLOWS =-
C  WORD 1 = THE TABLE NAME (LEFT=JUSTIFIED,BLANK=FILLED
C            HOLLERITH)
C  WORD 2 = UNUSED
C  WORD 3 = NO. OF ENTRIES IN THE TABLE
C NEXT FOLLOW TRIPLETS OF WORDS FOR EACH NAMELIST VARIABLE =
C  WORD 1 = VARIABLE NAME (SAME AS TABLE NAME)
C  WORD 2 = VARIABLE LOCATION (RIGHT JUSTIFIED,RELATIVE
C TO BEGINNING OF CODE FIELD=LENGTH)
C  WORD 3 = SUBSCRIPT INFORMATION
C  = IF ZERO, THE VARIABLE IS SINGLY SUBSCRIPTED
C  = IF NON-ZERO, THE WORD IS INTERPRETED AS BEING
C MADE UP OF THREE 18 BIT FIELDS, DENOTING
C THE RIGHTMOST BIT AS 0, THE THREE FIELDS ARE
C BITS 0=17, 18=35, AND 36=53, RESPECTIVELY,
C THESE THREE FIELDS CONTAIN NSUB1, NSUB2, AND
C NSUB3, RESPECTIVELY.
C THE LENGTH OF THE TABLE SHOULD BE 3*(WORD(3)+1) LOCATIONS,
C WRITTEN BY J.L. NORTON, LASL T=3, 1974
C INTEGER TABLE, SHIFT, AND, OR, COMP
C DIMENSION TABLE(3, 1), FMT(10)
C COMMON/ERRORC/ IERRT
C IEFLAG=0
C GET NUMBER OF ENTRIES CURRENTLY IN THE TABLE
C NENTRY=TABLE(3, 1)
C COMPUTE SUBSCRIPT FOR NEW TABLE ENTRY
C ISUB=ENTRY+2
C MAKE SURE THERE IS STILL TABLE SPACE LEFT

170
55 C IF(ISUB,LE,TABLE(2,1)) GO TO 10
56 C NO, FATAL ERROR.
57 C IFFLAG=5
58 C GO TO 999
59 C 10 CONTINUE
60 C SET UP TABLE ENTRY
61 C TABLE(1,ISUB)=NAME
62 C TABLE(2,ISUB)=LOCF(LOCA)
63 C TABLE(3,ISUB)=0
64 C COMPOSE SUBSCRIPT WORD IF VARIABLE IS AN ARRAY
65 C IF(NSUB,EQ,0) GO TO 30
66 C ARRAY HAS ONLY ONE SUBSCRIPT, DENOTE THIS BY SETTING SUBSCRIPT
67 C FIELD 1 TO 1 AND LEAVING ALL THE REST ZERO.
68 C TABLE(3,ISUB)=1
69 C GO TO 30
70 C ARRAY HAS MORE THAN ONE SUBSCRIPT
71 C 20 CONTINUE
72 C SEE IF AT LEAST THE FIRST SUBSCRIPT IS SPECIFIED, IF NOT,ERROR.
73 C IF(NSUB1.LE,0) GO TO 40
74 C FIRST SUBSCRIPT PRESENT, STORE IT AND SEE IF ARRAY HAS MORE THAN
75 C TWO SUBSCRIPTS.
76 C TABLE(3,ISUB)=NSUB1
77 C IF(NSUB1,EQ,2) GO TO 30
78 C YES, THE SECOND SUBSCRIPT MUST BE SPECIFIED, IF NOT,ERROR.
79 C IF(NSUB2.LE,0) GO TO 50
80 C SECOND SUBSCRIPT PRESENT, STORE IT AND SEE IF ARRAY HAS MORE THAN
81 C THREE SUBSCRIPTS.
82 C TABLE(3,ISUB)=OR(TABLE(3,ISUB),SHIFT(NSUB2,10))
83 C IF(NSUB2,EQ,3) GO TO 30
84 C YES, THE THIRD SUBSCRIPT MUST BE SPECIFIED, IF NOT,ERROR.
85 C IF(NSUB3.LE,0) GO TO 60
86 C
SUBROUTINE NXTCOL(IF,IEFLAG)

ROUTINE TO GET THE NEXT CHARACTER FROM THE CURRENT NAMELIST CARD
AND, IF NECESSARY, READ ANOTHER CARD FROM FILE IF

WRITTEN BY J. L. NORTON, LASL T-30197, 1974

COMMON/ARRCON/ICHAR(80), ISUB, ITEST, NSUB, NSUBV(4), IENTRY, ISBSPT,
CREAL, CIMAG, CMPLEX, DIM, MULTSV, IVSUM, MUL, LIST, HOLLER, DOBLE
LOGICAL LIST, HOLLER

INCREMENT THE COLUMN NO.

ISUB = ISUB + 1

PICK UP THE CHARACTER IN COLUMN ISUB
C     ITEST=ICHAR(ISUB)
C     IF ISUB HAS NOT MOVED ACROSS A CARD BOUNDARY, WE ARE ALL DONE
C     IF(ISUB,LE,80) GO TO 30
C     MUST READ NEXT CARD, IFLAG WILL BE RETURNED NON-ZERO ONLY IF
C     AN EOF WAS ENCOUNTERED.
C     CALL READIT(IF,IFLAG)
C     IF(IFLAG,EQ,0) GO TO 10
C     AN EOF WAS READ, WE ARE ALL THROUGH.
C     CALL MESSOT(32)
C     IFLAG=-32
C     RETURN
C 10 CONTINUE
C     READ WAS SUCCESSFUL, RESET COLUMN AND CHARACTER,
C     ISUB=2
C     ITEST=ICHAR(ISUB)
C     IF PROCESSING A HOLLERITH FIELD, NO MORE CHECKING
C     IF(HOLLER) GO TO 20
C     CHECK FOR $ IN CC2, IF NOT, ALL FINISHED CHECKING.
C     IF(ISET,NE,1HS) GO TO 20
C     $ IN CC2, PROBABLY BEGINNING OF NEXT NAMELIST STATEMENT, MUST
C     HAVE HAD MISSING TERMINAL $ ON PREVIOUS CARD, GO BACK AND
C     READ THE CARD IN ERROR.
C     BACKSPACE IF
C     BACKSPACE IF
C     CALL READIT(IF,IFLAG)
C     SET THE COLUMN POINTER TO CC80, GO PRINT ERROR MESSAGE, AND QUIT
C     ISUB=80
C     CALL MESSOT(33)
C     IFLAG=33
C     RETURN
C 20 CONTINUE
C     EVERYTHING CHECKED OUT, SET COLUMN TO 1 AND PICK UP THE
C     CHARACTER IN CC1.
C     ISUB=1
C     ITEST=ICHAR(ISUB)
C     IF FLAG IS SET, PRINT THE LAST CARD READ
SUBROUTINE READIT(IF,IEFLAG)

C ROUTINE TO READ THE NEXT CARD ON FILE IF

C WRITTEN BY J.L. NORTON, LASL T=5, 1974

C COMMON/ARRCON/ICHAR(50), ISUB, ITEST, NSUBV(4), IENTRY, ISBSPT,
C 1 CREAL, CIMAG, CMPLEX, DIM, MULTSV, IVSUM, MULT, LIST, HOLLER, DOBLE

C READ THE NEXT CARD

C READ(IF,30) ICHAR

C CHECK FOR END-OF-FILE

C IF(EOF,IF) 10, 20

C YES, SET THE FLAG,

C 10 CONTINUE

C IFLAG=1

C 20 CONTINUE

C RETURN

C 25 30 FORMAT(50A1)

C END

SUBROUTINE OPDMP(FWA, LWA, IFILE)

C ROUTINE TO DUMP SMALL CORE MEMORY IN CDC COMPASS MNEUMONICS

C FWA = FIRST WORD TO BE DUMPED

C LWA = LAST WORD TO BE DUMPED

C IFILE = FILE TO WHICH TO DIRECT OUTPUT

C WRITTEN BY J.L. NORTON, LASL T=5, 1975

C INTEGER FWA, FWA, GETIT, SHIFT, AND, OR

C DIMENSION IDIGIT(20), ICODE(4), IREG(4), IOPRN(13, 4), IDNAME(8)

C DIMENSION IOP(42), IHEG(20)
DATA IO/P,2HPS,2HL,2HML,2HMJ,2HRX,2HMX,2HRI,2HTB,2HIB,2HRO,2HOB,
1 2HRJ,2HPJ,2HRZ,2HNN,2HPL,2HNG,2HR,G,2HDF,2HID,2HES,2HME,2HGE,
2 2HLT,2HBP,2HLX,2MXX,2HMX,2MNX,2MXZ,2HUX,2HFX,2HDF,2HDX,2HRX,2HIX,
3 2HNO,2HCH,2H8A,2H8B,2H8X/
16 DATA IDNAME/IH0,1H1,1H2,1H3,1H4,1H5,1H6,1H7/
19 C  
20 C  IF FWA,GT,LWA,IGNORE THE CALL
21 C  IF (FWA,GT,LWA) RETURN
23 C  FWAP=FMA
24 C  LOOP OVER ALL WORDS TO BE DUMPED
25 C  10 CONTINUE
26 C  10 CONTINUE
27 C  GET THE CONTENTS OF THE CURRENT ADDRESS (FWAP) TO BE DUMPED
28 C  IWORD=GETIT(FWAP)
29 C  BREAK THE WORD UP INTO INDIVIDUAL DIGITS
30 C  DO 20 I=1,20
31 C  IDIGIT(I)=AND(sHIPT(IWORD),7B)  
32 C  CONTINUE
33 C  COMPUTE HOLLERITH EQUIVALENT OF DIGITS
34 C  DO 30 I=1,20
35 C  ISUB=IDIGIT(I)+1
36 C  IHEO(I)=IDNAME(ISUB)  
37 C  CONTINUE
38 C  IDN IS THE CURRENT DIGIT NO, OF THE WORD BEING PROCESSED
39 C  IDN=1
40 C  IPART IS THE NO, OF THE INSTRUCTION (1-4) BEING PROCESSED
41 C  FOR THE CURRENT WORD
42 C  IPART=1
43 C  INITIALIZE THE OUTPUT FIELDS
44 C  DO 50 I=1,4
45 C  ICODE(I)=2H
46 C  IREG(I)=1H
47 C  CONTINUE
48 C  LOOP OVER ALL DIGITS
49 C  DO 50 CONTINUE
50 C  LOOP OVER ALL DIGITS
51 C  IOPNO IS THE NO, OF THE CHARACTER OF THE OPERAND FIELD CURRENTLY
52 C  BEING FILLED
53 C
71  IOPNO=1
72  IGDIT=IDIGIT(IDN)+1
73  GO TO (70,560,440,500,520,620,620,620),IGOT
74  C  FIRST DIGIT OF OP IS 0
75  C  CONTINUE
76  IDN=IDN+1
77  IGDIT=IDIGIT(IDN)+1
78  GO TO (60,190,330,340,350,350,350,350),IGOT
79  C  OP IS PS (00)
80  C  CONTINUE
81  ITYPE=1
82  C  CHECK FOR ERRORS
83  C  IF(IDN.LE.12) GO TO 100
84  C  SEE IF THIS IS END OF WORD PADDED WITH ZEROS, IF SO,
85  C  PROCESS LIKE NO.
86  C  IF(IDN.EQ.17) GO TO 600
87  C  WORD IS IN ERROR, MUST NOT BE INSTRUCTION, PRINT MESSAGE AND GO ON
88  C  TO NEXT WORD.
89  C  CONTINUE
90  C  WRITE(IFILE,760) FWAP,IPNDR
91  C  GO TO 100
92  C  CONTINUE
93  ITYPE=1
94  C  PROCESS INSTRUCTION OF THE FORM OP K
95  C  IDN=IDN+2
96  C  PROGRESS K
97  C  CONTINUE
98  ISTD=0
99  DO 120 I=-1,6
100  IDN=IDN+1
101  ISTD=STD(IDN,SHIFT(IDIGIT(IDN),3*(6-I)))
102  CONTINUE
103  ISTD=STD(SHIFT(IDST,42),42)
104  IF(IDST.GE.0) GO TO 130
105  ISTD=STD
106  IOPRN(IOPNO,IPART)=1H=
107  GO TO 140
108  CONTINUE
109  IOPRN(IOPNO,IPART)=1H=
110  CONTINUE
111  ISTD=STD
112  DO 120 I=-1,6
113  IDN=IDN+1
114  ISTD=STD(IDN,SHIFT(IDIGIT(IDN),3*(6-I)))
115  CONTINUE
116  ISTD=STD(SHIFT(IDST,42),42)
117  IF(IDST.GE.0) GO TO 130
118  ISTD=STD
119  IOPRN(IOPNO,IPART)=1H=
120  GO TO 140
121  CONTINUE
122  IOPRN(IOPNO,IPART)=1H=
123  CONTINUE
124  ISTD=STD
125  DO 150 I=-1,6
126  IDN=IDN+1
127  IDN=AND(SHIFT(IDST,3*(6-I)),7B)
IOPRN(IOPNO,IPART)=IDNAME(ID+1)
IOPNO=IOPNO+1
150 CONTINUE
C
151 C FINISH PARCEL
152 C
153 160 CONTINUE
154 ICODE(IPART)=IOP(ITYPE)
155 IF(IDN,GE,20) GO TO 170
156 IPART=IPART+1
157 IDN=IDN+1
158 IF(IDN,GT,16) GO TO 90
159 GO TO 60
160 C
161 C WORD IS FINISHED, PRINT THE LINE.
162 C
163 170 CONTINUE
164 WRITE(IFILE,770) FWAP,INWORD,(ICODE(I),IREG(I),IOPRN(J,I),J=1,13), 1 1=1,IPART)
165 C
166 C SEE IF THIS WAS THE LAST WORD
167 C
168 180 CONTINUE
169 IF(FWAP,GE,LWA) RETURN
170 C
171 C NO, CONTINUE.
172 C
173 FWAP=FWAP+1
174 GO TO 10
175 C FIRST TWO DIGITS OF OP ARE 01
176 C
177 190 CONTINUE
178 IDN=IDN+1
179 IGOT=IDIGIT(IDN)+1
180 GO TO (200,210,210,230,240,240,250,310),IGOT
181 C
182 C OP IS RJ (010)
183 C
184 200 CONTINUE
185 ITYPE=12
186 IDN=IDN+1
187 GO TO 110
188 C
189 C OP IS 011 OR 012
190 C
191 210 CONTINUE
192 ITYPE=IGOT
193 C PROCESS INSTRUCTION OF THE FORM OP BJ*K
194 C
195 220 CONTINUE
196 IF(IDN,GT,13) GO TO 90
197 IDN=IDN+1
198 IOPRN(1,IPART)=1MB
199 IOPRN(2,IPART)=IHEQ(IDN)
200 IOPNO=3
GO TO 110
C OP IS MJ
C CONTINUE
IF(IDENTION+1).NE.0, OR, IDENTION+2), NE, 0) GO TO 220
IF(IDENTION.GT.15) GO TO 90
IDENTN+7
GO TO 160
C OP IS 014 OR 015
C CONTINUE
C PROCESS INSTRUCTION OF THE FORM OPJ XK
C IDENTN+1
C IREG(IPART)=IHEQ(IDENTN)
C IDEN+1
C IOPRN(1,IPART)=1MX
C IOPRN(2,IPART)=IHEQ(IDENTN)
C GO TO 160
C OP IS 016
C CONTINUE
IF(IDENTION+1), NE, 0) GO TO 280
C PROCESS INSTRUCTION OF THE FORM OP BK
C CONTINUE
C IDENTN+2
C CONTINUE
C IOPRN(1,IPART)=1HB
C IOPRN(2,IPART)=IHEQ(IDENTN)
C GO TO 160
C CONTINUE
IF(IDENTION+2), NE, 0) GO TO 290
C PROCESS INSTRUCTION OF THE FORM OPJ
C C IDENTN+1
C C IREG(IPART)=IHEQ(IDENTN)
C C IDENTN+1
C C GO TO 160
C CONTINUE
C PROCESS INSTRUCTION OF THE FORM OPJ BK
C C C IDENTN+1
C C C IREG(IPART)=IHEQ(IDENTN)
C C C IDENTN+1
242 GO TO 270
243 C OP IS 017
244 C
245 310 CONTINUE
246 IF(IDIGIT(IDN+1);NE,0) GO TO 320
247 ITYPE=10
248 GO TO 260
250 320 CONTINUE
251 ITYPE=11
252 GO TO 300
253 C OP IS JP (02)
254 C
255 330 CONTINUE
256 ITYPE=13
257 C PROCESS INSTRUCTION OF THE FORM OP BJ+K
258 C
259 IF(IDN,GT,12) GO TO 90
260 IDN=IDN+1
261 IPRRN(1,IPART)=1HB
262 IPRRN(2,IPART)=IMEQ(IDN)
263 IDN=IDN+1
264 IPRRN=3
265 GO TO 110
266 C
267 340 CONTINUE
268 IF(IDN,GT,12) GO TO 90
269 IDN=IDN+1
270 IPRRN=1
271 IPRRN(2,IPART)=1H
272 IPRRN(3,IPART)=IMEQ(IDN)
273 IPRRN(4,IPART)=1H,
274 IPRRN=4
275 GO TO 110
276 C
277 350 CONTINUE
278 IF(IDN,GT,12) GO TO 90
279 ITYPE=I50+17
280 C PROCESS INSTRUCTION OF THE FORM OP BJ, BJ,K
281 C
282 IPRRN(1,IPART)=1HB
283 IPRRN(2,IPART)=IMEQ(IDN)
284 IPRRN(3,IPART)=1H,
285 IPRRN(4,IPART)=1HB
FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.

FIRST DIGIT OF OP IS 1, OP IS B#X.
420 CONTINUE
C PROCESS INSTRUCTION OF THE FORM OPI
421
C
422 CONTINUE
IDPRN1(IPRN0,IPART)=1M/
GO TO 390
C
430 CONTINUE
IDN=IDN+1
IREG(IPART)=IHEQ(IDN)
IDN=IDN+1
KREG=IHEQ(IDN)
IDPRN(1,IPART)=1M=
IDPRN(2,IPART)=1HX
IDN=IDN+1
IDPRN(3,IPART)=IHEQ(IDN)
IF(IGOT,EQ,5) GO TO 160
IGOT=IGOT=5
IDPRN4=4
GO TO (380,400,410),IGOT
C
C FIRST DIGIT OF OP IS 2
C
440 CONTINUE
IDN=IDN+1
IGOT=IDIGIT(IDN)+1
GO TO (450,460,450,460,490,490,490,490,490,490),IGOT
C
C OP IS LXI
C
450 CONTINUE
ITYPE=27
IF(IGOT,EQ,1) GO TO 470
C
C PROCESS INSTRUCTION OF THE FORM OPI BJ,XK
C
460 CONTINUE
IDN=IDN+1
IREG(IPART)=IHEQ(IDN)
IDN=IDN+1
IDPRN(1,IPART)=1HB
IDPRN(2,IPART)=IHEQ(IDN)
IDN=IDN+1
IDPRN(3,IPART)=1H,
IDPRN4,IPART)=1HX
IDPRN(5,IPART)=IHEQ(IDN)
GO TO 160
C
C PROCESS INSTRUCTION OF THE FORM OPI JK
C
470 CONTINUE
IDN=IDN+1
IREG(IPART)=IHEQ(IDN)
IDN=IDN+1
IDPRN(1,IPART)=IHEQ(IDN)
IDN=IDN+1
IDPRN(2,IPART)=IHEQ(IDN)
IDPRN(3,IPART)=1HB
GO TO 160
C
C OP IS AXI

480 CONTINUE
ITYPE=28
IF(IGOT,EQ,2) GO TO 470
GO TO 460
C
C OP IS MXI,ZXI,UXI,OR PXI

490 CONTINUE
ITYPE=IGOT+25
GO TO 460
C
C FIRST DIGIT OF OP IS 3

500 CONTINUE
IDN=IDN+1
IGOT=IDIGIT(IDN)
ITYPE=IGOTP/2+54
IF(MOD(IGOTP,2),NE,0) GO TO 510
IGOT=3
GO TO 370
C
C FIRST DIGIT OF OP IS 4

510 CONTINUE
IGOT=4
GO TO 370
C
C OP IS FLOATING MULTIPLY

520 CONTINUE
ITYPE=34
GO TO 560
C
C OP IS MXI OR CXI

530 CONTINUE
ITYPE=35
GO TO 560
C
C

540 CONTINUE
ITYPE=29
IF(IGOT.EQ.7) ITYPE=39
GO TO 470

C
C OP IS FLOATING DIVIDE
C
580 CONTINUE
ITYPE=34
IF(IGOT.EQ.5) ITYPE=36
IGOT=5
GO TO 370
C
C OPP IS NO
C
590 CONTINUE
ITYPE=38

C PROCESS IS 15 BIT INSTRUCTION OF THE FORM OP N
C
600 CONTINUE
IDN=IDN+1
IF(DIGIT(IDN).NE.0.OR.DIGIT(IDN+1).NE.0.OR.DIGIT(IDN+2).NE.0)
1 GO TO 610
IDN=IDN+2
GO TO 160
610 CONTINUE
IOPRN(1,IPART)=IHEQ(IDN)
IOPRN(2,IPART)=IHEQ(IDN+1)
IOPRN(3,IPART)=IHEQ(IDN+2)
IOPRN(4,IPART)=IHB
IDN=IDN+2
GO TO 160
C
C OP BEGINS WITH 5,6, OR 7
C
620 CONTINUE
ITYPE=IGOT+34
IDN=IDN+1
IGOT=DIGIT(IDN)+1
IF(IGOT,LT.4.AND.IDN,GT.12) GO TO 90
IDN=IDN+1
IREG(IPART)=IHEQ(IDN)
IDN=IDN+1
JREG=IHEQ(IDN)
IF(IGOT,LT.4) GO TO 630
IDN=IDN+1
KREG=IHEQ(IDN)
630 CONTINUE
GO TO (640,660,670,680,700,710,730,750),IGOT
640 CONTINUE
IOPRN(1,IPART)=IHA
650 CONTINUE
IOPRN(2,IPART)=IREG
IOPR0#3
GO TO 110
660 CONTINUE
IOPRN(1,IPART)=IHB
GO TO 650
SUBROUTINE PABORT
C
C ROUTINE TO HANDLE ABORT EXCHANGE PACKAGE DUMPS
C WRITTEN BY J.L. NORTON, LASL T=3, 1974
C
INTEGER SHIFT
LOGICAL FILM
COMMON/YBCS/RESTR, FILM, PAPER, IPD, IFD
COMMON/IEDMP/IDUMP(16)
C
CALL SECOND(TJLN)
DO 10 IPX=6, IFD, 6
10 WRITE(IPX, 30) TJLN

C
CONTINUE
IOPRN(1, IPART) *1HX
GO TO 650
650 CONTINUE
IOPRN(3, IPART) *1HX
IOPRN(3, IPART) *1HA
GO TO 690
CONTINUE
IOPRN(1, IPART) *1HA
690 CONTINUE
IOPRN(3, IPART) *1HB
IOPRN(4, IPART) *1HB
IOPRN(5, IPART) *1HB
GO TO 160
CONTINUE
IOPRN(3, IPART) *1HB
700 CONTINUE
IOPRN(3, IPART) *1HB
IOPRN(1, IPART) *1HB
IOPRN(5, IPART) *1HB
GO TO 160
60 TO 680
CONTINUE
IOPRN(3, IPART) *1HB
GO TO 690
CONTINUE
IOPRN(3, IPART) *1HB
IOPRN(4, IPART) *1HB
CONTINUE
IOPRN(1, IPART) *1HB
740 CONTINUE
IOPRN(3, IPART) *1HB
GO TO 740
750 CONTINUE
IOPRN(3, IPART) *1HB
GO TO 740
CONTINUE
IOPRN(1, IPART) *1HB
760 FORMAT(1H ,06, 3X020, 3X8(1H=),
1 54WORD CONTAINS ILLEGAL INSTRUCTION(8), MUST BE DATA OR ,
2 H0CLOBBERED, 8(1H=))
CONTINUE
IOPRN(3, IPART) *1HB
GO TO 740
770 FORMAT(1H ,06, 3X020, 4(3XA2,A1,3X13A1))
END
PRINT THE HARDWARE EXCHANGE PACKAGE

DO 20 IPX=6,IPD,6
20 WRITE(IPX,40)
CALL DMPPK(6,IDUMP)
SEND IT TO FILM IF FILM IS "TRUE."

IF(FILM) CALL DMPPK(12,IDUMP)
ENTRY PDMPK
GET THE P COUNTER FOR THE ABORT

IFW=SHIFT(IDUMP,*36)
SUBTRACT 32 (40 OCTAL) FROM IT. IF RESULT IS NEGATIVE, SET IT TO ZERO.

IFW=IFW-32
IF(IFW,LT,0) IFW=0
ADD 37 OCTAL (IFW=40B+77B) TO P COUNTER, IF RESULT IS OUT OF RANGE, SET IT TO 77B.

IFW=IFW+77B
IF(IFW,GT,150077B) IFW=0
IF(IFW,GT,150077B) IFW=77B
PRINT OP CODE DUMP AROUND ABORT ADDRESS

call CDPDMP(IFW,IFWB,6)
SEND OP CODE DUMP TO FILM IF FILM IS "TRUE.

IF(FILM) CALL CDMPD(IFW,IFWB,12)
RETURN
30 FORMAT(1HI,"TIME AT ABORT WAS F9.3")
40 FORMAT(//1H,10(1H+),25H HARDWARE EXCHANGE PACKAGE,10(1H+))
END

SUBROUTINE PARPLT
ROUTINE TO PLOT PARTICLES

ORIGINALLY WRITTEN BY A.A. AMSDEN, LASL T=3
MODIFIED AND DOCUMENTED BY J.L. NORTON, LASL T=3, 1975

* ---- BEGIN COMDECK PARAM ----
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
COMMON/XTENC/XTEN,YTEN
DATA BCD/1H /
C
ADVANCE FILM TO NEXT FRAME
CALL ADV(1)
C
INITIALIZE PLOT VALUES
IPXL=65
IPXR=660
IPYB=900
FIPXL=65,
FIPXR=660,
FIPYB=900,
YUP=TOP+2,+PRITE
YLB=BOTM-3,+PRITE
PYCNVP=FLOAT(IPYT-IPYB)/(YUP-YLB)
PXCONV=PXCONV*PYCNVP/PYCNV
PYCONV=PYCNVP
SET UP CALL TO TICBOX
XCONVP=PXCONV
YCONVP=PYCONV
IXLSV=IXL
IXPSV=IXR
IYBSV=IYB
IYTSV=IYT
IXL=IPXL
IXR=IPXR
IYR=IPYR
IYT=IPYT
FIYRVS=FIYB
FIXLSV=FIXL
FIYH=FIYR
FIXL=FIXPL
DRAW AND LABEL THE PLOT FRAME
CALL TICBOX
LABEL THE PARTICLE PLOT
CALL LINCNT(68)
WRITE(IFD,100) PXR,PYR,PTY
WRITE(IFD,110) JNM,NAME,T,NCYC
LOOP OVER ALL THE PARTICLES AND PLOT THEM
IECP IS THE LCM INDEX
NPPT IS THE PARTICLE COUNTER
IECP=1
NPPT=0
BRING A BUFFER LOAD OF PARTICLE DATA FROM LCM INTO SCM
124  C  10 CALL ECRD(AASC, NLCP1+IECP=1, LPB, IDUM)
125  C  KP IS THE SCM INDEX
126  C  KP=1
127  C  SKIP THE PARTICLE IF ITS X-COORDINATE IS NEGATIVE
128  C  20 IF(XPAR(KP),LT.0.) GO TO 30
129  C  CALCULATE THE RASTER COORDINATES OF THE PARTICLE
130  C  IX1=FIPXL*( XPAR(KP)*PXL)*PXCONV
131  C  IY1=FIPYL*(YPAR(KP)*YLX)*PYCONV
132  C  PLOT IT UNLESS IT IS OUTSIDE OF THE PLOTTING RECTANGLE
133  C  IF((IX1,GT,IXR,OR. IX1,LT,IXL)) GO TO 30
134  C  IF((IY1,GT,IYR,OR. (IY1,LT,IY1))) GO TO 30
135  C  CALL PLT(IX1, IY1,42)
136  C  CALL PLT(IX1, IY1,42)
137  C  INCREMENT THE PARTICLE COUNTER AND SEE IF ALL HAVE BEEN PLOTTED
138  C  30 NPPT=NPPT+1
139  C  IF(NPPT.EQ.NPT) GO TO 40
140  C  NO, INCREMENT THE SCM POINTER AND SEE IF THE BUFFER NEEDS TO
141  C  BE REFILLED,
142  C  KP=KP+3
143  C  IF(KP,LT,LPB) GO TO 20
144  C  YFS, INCREMENT THE LCM INDEX AND GO REFILL THE SCM BUFFER,
145  C  IF(CP,IECP+LPB
146  C  GO TO 10
147  C  ALL PARTICLES HAVE BEEN PLOTTED, SEE IF ANY TIME-DEPENDENT
148  C  PLOTTING IS DESIRED,
149  C  40 CONTINUE
150  C  IF(IST,LE,1) GO TO 80
151  C  YES, SEE IF THERE HAVE BEEN AT LEAST TWO TIME PERIODS SAVED,
152  C  IF(NLNG,LT,2) GO TO 80
153  C  YES, PREPARE TO DO THE TIME-DEPENDENT PARTICLE PLOT,
154  C  CALL ADV(1)
155  C  CALL TICBOX
156  C  NIST=NPT/IST
157  C  NTOT=NPT/NIST
158  C  DO 70 I=1,NTOT
159  C  70
DO 60 J=1,NLNG
          SAVE THE LAST POSITION IN TIME
          I3=I2
          IY3=IY2
          IOK3=I0K2
          IBEGIN=NLCP1+NLCP2+2*(NP1*(J-1)+1)
          CALL ECRD(XTEN,IBEGIN,2,IDUM)
          IX2=FIPXL+(PAX(I,J)*XL)*PXCONV
          IY2=FIPYR+(PAY(T,J)-YLBI)*PYCONV
          IOK2=0
          IF(IX2,GT. IXR,OR. IX2.LT. IXL,OR. IY2,GT. IY1,OR. IY2.LT. IYT) IO2=1
C       THE INITIAL POSITION WITH A STAR
          IF(J. NE. 1) GO TO 50
          IF(IOK2.EQ.0) CALL PLT(IX2, IY2, 44)
GO TO 60
50 CONTINUE
          MARK PARTICLES WITH A DOT
          IF(IOK2.EQ.0) CALL PLT(IX2, IY2, 27)
          CONNECT THE PARTICLE WITH ITS NEIGHBOR IN TIME
          IF(IX2, EQ. 0, AND. IOK3, EQ. 0) CALL DRV(IX2, IY2, IX3, IY3)
          CONTINUE
60 CONTINUE
          LABEL THE PLOT
          CALL LINCNT(60)
          WRITE(IFD,90)
          WRITE(IFD,110) JNM, NAME, T, NCYC
10 CONTINUE
          RESTORE THE PLOT INDICES
          IXL=IXLSV
          IXR=IXRSV
          IYB=IYBSV
          IYT=IYTSV
          FYB=FYBSV
          FXL=FXLSV
          RETURN
C       FORHAT(25H TIME-DEPENDENT PARTICLES)
C       FORHAT(10H PARTICLES/1X5H PXR=1PE12, 5H PYB=12, 5H PYT=E12, 5)
C       FORHAT(1H ,4X410, 8A10,3X2HT, ,1PE12, 5, 1X6HCYCLE=E12, 5)
END
SUBROUTINE PCHCK

ROUTINE TO CHECK THE YAQUI LARGE AND SMALL CORE PARAMETERS

WRITTEN BY J.L. NORTON, LASL 1973, 1975

----- BEGIN COMDECK PARAM -----
COMMON/PCOM/NSCP1, ITARP, ITABXP, IPFB, NP1, NP2, NLCP1, NLCP2,
   1 NLCP3, NLCP4, IFMSZ
----- END COMDECK PARAM -----

LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAM, MU

COMMON/YSC1/AASC(NSCP1)
COMMON/YSC1/AASC(9600)
COMMON/YSC2/AA(1), ANC, AR, ADFAC, AOM, B0, COLAMU, CYL, DR, DT, DTC, DTFAC,
   1 DT0(10), DT0C(10), DT02, DT08, DT0POS, DTV, DZ, EM10, EPS, FIPXL, FIPXR,
   1 FIPYR, FIPYT, FJX, FJXY, FJY, FJYT, FREXR, GR, GROVEL, GZ, GZP, I, IBAR,
   1 IDT0, IJ, IJM, IP1, IP2, IP3, IPXY, IPYE, IPYT, IP1, IP2, IP3, ISC2, ISC3, ITV,
   2 JUNF, IXL, IXL, IYB, IYB, IYBAR
COMMON/YSC2/JCNJ, JP1, JP2, JP4, JUNF, JUNF02, KXI, LAM, LPR, MU, NAME(8),
   1 NCYX, NLC, NPS, NP4, NQ1, NQ1B, NQ12, NSC, NUMIT, ZORIG, DM, OMCYL, PXCONV
   2 F, PXL, PXR, PYB, PYCONV, PYT, RDT, REZON, REZSIE, REZYB, RIJBAR, RIBJB,
   3 FREY, FREZYT, ROMFXR, ROMFYb, I, THIRD, NCLST, TOUT, TWF
COMMON/YSC2/TUO, TUSI, NCQ, TNEG, TNEG, TUSV, TURB, TUP, PRIME, PHOTM,
   1 ING, NILNG, TP3, TUP0, TUALS, TK, T1, TUBENG, EP1, SAVI, SAVI, GLEVEL, TO, IST,
   2 VV, XCONV, XL, YR, YB, YCONV, YT, TPTOLD, DTSV, DTSLAST, F tyr, FYBD, YB0, YCNVLD,
   3 XCNVLD, FIXRO, FIXLO, IXRO, IXLO, ISVM, JSVM, OMN, OMX, WMX, WMX, JNM, T2, TLIM,
   4 ROMFXR, ROMFYB, JDUMP, TWHRD, TE, DTR, TMASS, DTSAV, DTCSAV, IDTV
   5 JTDV, PDTOY, IDTD, IDC, CIRC, TIS, POTE, UHOM, VDOM, TM, TGMX, TFM, JTM, ITG, JTG
   6 TTHAY, WMAYX, F, RMTNF, TSTRND
COMMON/YSC2/ZZ
COMMON/YSC4/ITAR(ITABP)
COMMON/YSC4/ITAB(1000)
COMMON/YSC5/RESTRT, FILM, PAPER, IPD, IFD
----- END COMDECK YAQSC ----- 

CHECK TO MAKE SURE THE SCM BUFFER WILL BE LARGE ENOUGH

IF(3*NQI.GT,NSCP1) CALL UNCLE(4,5HPCCHCK,33,
   1 33HNOT ENOUGH SMALL CORE FOR BUFFERS)

MAKE SURE LCM IS LARGE ENOUGH TO HOLD ALL THE ARRAYS

IF(NLC.GT.NLCP1) CALL UNCLE(4,5HSET,14,14HNOT ENOUGH LCM)
RETURN
END

SUBROUTINE PRTGEN
ROUTE TO GENERATE PARTICLES

ORIGINALLY WRITTEN BY A. A. AMSDEN, LASL T=3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T=3, 1974

----- BEGIN CODECK PARAM -----
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFMLS
----- END CODECK PARAM -----

----- BEGIN CODECK YSTORE -----

----- BEGIN CODECK YAUDMT -----

DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM,
1 (1), M(1), VM(1), P(1), PL(1), UP(1), UTIL(1), UTL(1), CQ(1), VTIL(1),
3 (1), ROL(1), AVXS(1), ROLX(1), SDL(1), DLS(1), CAPGM(1), TUG,
4 (1), SIG(1), TIS(1), GRRO(1), GRRPZ(1), TDVVEC(1), MTIL(1),
5 CONC(1), CETMP(1), ANCV(1), GRSV(1), GZSV(1), X13K(1), X24K(1),
6 Y13K(1), Y24K(1), XR13K(1), XZ24K(1), DKLS(1), AREA(1),
7

----- END CODECK YAUDMT -----
9 AASC(31), ANCV, (AASC(32), ANCV), (AASC(33), AVXSV, X13K), (AASC(34), AVXSV, X24K)
1

61 REAL M*, MP*, MPR, MTL
62 * ----- END COMDECK YADEQ ----- *
63 * ----- END COMDECK YSTORE ----- *
64 DIMENSION PARTN(3,7)
65 DATA (PARTN(IT), IT = 1, 21) / 21 *
66 DATA IEFLAG/0, NFE/N!
67 INTEGER COM, AND, OR
68 C
69 C SET UP THE NAMFIST INPIHT TABLE
70 C
71 ASSIGNED 170 TO IEWPT
72 CALL TARDEF(PARTN, SHPARTN, 7, IEWT)
73 CALL TARSET(PARTN, SHORPAR, DRPAR, IEFLAG, P, P, P, P)
74 CALL TARSET(PARTN, SHZPAR, DZPAR, IEFLAG, P, P, P, P)
75 CALL TARSET(PARTN, SHXC, XC, IEFLAG, P, P, P, P)
76 CALL TARSET(PARTN, SHYD, XD, IEFLAG, P, P, P, P)
77 CALL TARSET(PARTN, SHYD, XD, IEFLAG, P, P, P, P)
78 CALL TARSET(PARTN, SHYD, XD, IEFLAG, P, P, P, P)
79 C
80 C INITIALIZE PARTICLE COUNT
81 C
82 NPT = 0
83 C
84 C DEFINE CONSTANT
85 C
86 NQI = NQI * 2
87 C
88 LBP IS THE LENGTH OF ONE SMALL CORE BUFFER ROUNDED DOWN TO THE
89 NEAREST MULTIPLE OF THREE, PARTICLE DATA WILL BE STORED IN
90 A SMALL CORE BUFFER IN THREE WORD BLOCKS.
91 C
92 LBP = NQI / 3 * 3
93 C
94 KP IS THE SCM BUFFER SUBSCRIPT
95 C
96 KP = 1
97 C
98 C IECP IS THE CURRENT LCM READ ADDRESS
99 C
100 C IECP = 1
101 C
102 C IF THE MESH IS VARIABLE, RECALCULATE THE PARTICLE DR AND DZ
103 C
104 C IF (FREZXR.EQ.1, AND, FREZYT.EQ.1, AND, FREZYB.EQ.1) GO TO 10
105 C
106 C IT IS, COMPUTE XMAX, YMAX, AND YMIN BASED ON GEOMETRIC PROGRESSION
107 C RELATIONS, THE FIRST IUNF CELLS ARE OF WIDTH DR AND CONTRIBUTE
108 C IUNF*DR TO XMAX, THIS INCLUDES CELLS I=1 TO I=IUNF, CELL
109 C IUNF+1 HAS WIDTH FREZ*DR, CELL IUNF+2 HAS WIDTH FREZ*(FREZ*DR),
110 C ETC, THE LAST CELL I=IBAR WILL HAVE WIDTH FREZ**((IBAR-IUNF)*DR)
111 C AND THE SUM WILL BE THE PROGRESSION
112 C FREZ*DR + FREZ**2*DR + , , , + FREZ**(IBAR-IUNF)*DR
113 C = FREZ*(1 + FREZ + FREZ**2 + , , , + FREZ**(IBAR-IUNF-1)) * DR
114 C = FREZ*DR*(1-FREZ**((IBAR-IUNF)))/(1-FREZ)
115 C
116 C
117 C
118 C
119 C
120 C
121 C
122 C
123 C
124 C
125 C
YMAX AND YMIN ARE COMPUTED IN A SIMILAR FASHION.

IF(FREZXR,GT,1.) XMAX=(FLOAT(IUNF)+FREZXR*(1.-FREZXR***(IBAR-IUNF))
1.*ROMFXR)*DR
IF(FREZYT,GT,1.) YMAX=FREZYO*(FLOAT(JUNF02)+FREZYT*(1.-FREZYT***(
1.JCEN=JUNF02)*ROMFYT)*DZ
IF(FREZYB,GT,1.) YMIN=FREZYO*(FLOAT(JUNF02)+FREZYB*(1.-FREZYB***(
1.JCEN=JUNF02)*ROMFYB)*DZ

READ IN QUANTITIES FOR ONE PARTICLE REGION, FIRST SET DEFAULT VALUES.

10 CONTINUE

DRPAR IS THE R DIMENSION OF THE RECTANGLE IN THE MIDDLE OF WHICH
ONE PARTICLE WILL BE PLACED

ZPAR IS THE Z RECTANGULAR DIMENSION

X, Y, X, AND Y ARE THE DIMENSIONS OF THE ENTIRE PARTICLE REGION,
IF THE REGION IS RECTANGULAR, (X, Y) IS THE LOWER LEFT-HAND
CORNER AND (X, Y) IS THE UPPER RIGHT-HAND CORNER, IF THE
REGION IS CIRCULAR, X=0, Y=0, X IS UNUSED, Y IS THE Z VALUE OF
THE CENTER OF THE CIRCLE (ASSUMED TO BE ON THE AXIS OF
SYMMETRY), AND X IS THE RADIUS OF THE CIRCLE.

X=0,
Y=0,
X=0,
Y=0.

DO THE ACTUAL READ

CALL NMLST(PARTN,5,IEFLAG)

CHECK FOR INPUT ERRORS

IF(IEFLAG,NE,0) CALL UNCLE(4,6HPRTGEN,26,
1.26HPRTN NMLST INPUT ERROR)

SEF IF THIS WAS THE LAST PARTICLE REGION, IF SO WE ARE ALL DONE.

IF(DRPAR.LE.0.) GO TO 140

NO CHECK FOR INPUT ERRORS.

IF(DZPAR.LE.0.) CALL UNCLE(4,6HPRTGEN,14,14ERROR IN DZPAR)
IF(YD,LE.0.) CALL UNCLE(4,6HPRTGEN,11,11ERROR IN XD)

NO ERRORS, PRINT OUT THE INPUT VARIABLES.

DO 20 IPX=6,1F0,6

PRINT(IPX,1AM) DRPAR,DZPAR,XC,YC,XD,YD
SEE IF PARTICLE REGION IS RECTANGULAR OR CIRCULAR

IF(YO, EQ, 0, ) GO TO 40

REGION IS RECTANGULAR, SET PARTICLE REGION BOUNDS. IF VARIABLE ZONING IS BEING USED, PARTICLE REGION BOUNDS ARE SET TO PROBLEM BOUNDS.

IF(FREXR, EQ, 1, , AND, FREYY, EQ, 1, , AND, FREYZ, EQ, 1, ) GO TO 30

VARIABLE ZONING

PTOP = YMAX
PROTM = YMIN
PRITE = XMAX
PLEFT = O,
GO TO 50

UNIFORM ZONING

30 CONTINUE

PTOP = YD
PROTM = YC
PRITE = XD
PLEFT = XC
GO TO 50

REGION IS CIRCULAR

40 CONTINUE

PTOP = YC*XD
PROTM = YC*XD
PRITE = XD
PLEFT = XC
R2 = XD**2
50 CONTINUE

INITIALIZE THE OLD PTOP

PTPOLD = PTOP

BEGIN LOOP OVER PARTICLES

INITIALIZE THE Y COORDINATE

YTE = PROTM + 5*DZPAR

60 CONTINUE

INITIALIZE THE X-COORDINATE

XTE = PLEFT + 5*DRPAR

70 CONTINUE

IF PARTICLE REGION IS CIRCULAR, AN EXTRA CHECK MUST BE MADE TO SEE IF THE PARTICLE IS IN THE REGION
230 IF(YD,NE.,0.) GO TO 80
231 IF((YTE+YC)**2+XTE**2.GT.R2) GO TO 100
232 R2 CONTINUE
233 C
234 C STORE THE PARTICLE COORDINATES AND THE PARTICLE MASS
235 C
236 XPAR(KP)=XTE
237 YPAR(KP)=YTE
238 MPAR(KP)=M.
239 C
240 C INCREMENT THE SCM BUFFER SUBSCRIPT AND THE PARTICLE COUNT
241 C
242 KP=KP+3
243 NPT=NPT+1
244 C
245 C SEE IF THE SCM BUFFER NEEDS TO BE FLUSHED
246 C
247 IF(KP.GT.LPBO) GO TO 110
248 C
249 C NO, INCREMENT THE PARTICLE X-COORDINATE.
250 C
251 90 XTE=XTE+DZPAR
252 C
253 C SEE IF WE HAVE GONE OUTSIDE OF THE PARTICLE REGION
254 C
255 IF(XTE.LE.PRITE) GO TO 70
256 C
257 C YES, INCREMENT THE PARTICLE Y-COORDINATE.
258 C
259 100 YTE=YTE+DZPAR
260 C
261 C SEE IF Y HAS GONE OUTSIDE THE PARTICLE REGION
262 C
263 IF(YTE.LE.PTOP) GO TO 60
264 C
265 C YES, WE ARE DONE WITH THIS PARTICLE REGION, GO READ IN ANOTHER
266 C CARD.
267 C
268 C GO TO 10
269 C
270 C FLUSH THE BUFFER TO LCM IF THERE IS ROOM
271 C
272 110 CONTINUE
273 C
274 120 CALL UNCLE(A,6HPRTGEN,25,25HPARTICLE ARRAY OVERFLOWED)
275 C
276 CALL ECWR(AASC,NLCPl+LPB-1,LPB,IDUM)
277 C
278 C INCREMENT THE LCM ADDRESS,RESET THE SCM SUBSCRIPT,AND PROCEED
279 C
280 IECP=IECP+LPB
281 KP=1
282 GO TO 90
283 C
284 C ALL DONE, FLUSH THE SCM BUFFER TO LCM IF THERE IS ROOM.
285 C
286 140 CONTINUE
Routines PRTOV

ORIGINALLY WRITTEN BY A. A., AM8DEN, LASL, T-M 3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-M 3, 1975

ROUTINE TO MOVE PARTICLES

--- BEGIN COMDECK PARAM ---

1 COMMON/PSCOM/NSCP1, IPTAP, ITABX, ITABY, IPFB, NPT, NP2, NLCP1, NLCP2,
2 NLCP3, NLCP6, IFHMSZ ---

--- BEGIN COMDECK YSTORE ---

1 DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), HPAR(1), U(1), V(1), DELSH(1),
2 1, V(1), HPAR(1), MP1(1), DELSH(1), HPAR(1), RMP(1), RSCQ(1), E(1), ETIL(1), RVOL
3 (1), M1(1), R1(1), N1(1), PL1(1), UP1(1), UTIL1(1), UL1(1), CTIL(1)
4 3, V1(1), ROL1(1), AVVSY(1), AYVSY(1), DLSMO(1), DLSRO(1), DLSRO(1), CAPGAM(1), TEO
5 4, SIG8(1), TUS(1), GROR(1), GRRO(1), GRRO(1), TUG(1), NTIL(1),
6 4, CONC1(1), CTMP(1), ANC1(1), ANCV1(1), GBV(1), GBV(1), X13K(1),
7 6, Y13K(1), Y24K(1), X13K(1), X24K(1), DKLH(1), AREA(1)
8 --- END COMDECK YAQDIM ---

--- BEGIN COMDECK YAQ3C ---

LOGICAL RESTR, FILM, PAPER, TURB

REAL LAM, HU

--- BEGIN COMDECK YAQ3 ---

COMMON/VSC1/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC2/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC3/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC4/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC5/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC6/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC7/AASC(NSCP1)

--- BEGIN COMDECK YAQ8 ---

COMMON/VSC8/AASC(NSCP1)
DECIDE WHETHER TO ADD ANOTHER POINT TO THE TIME-DEPENDENT

PARTICLE DATA

IF(NCYCLE.EQ.1) GO TO 19
IF(T+EMP.LT,TOUT) GO TO 20

IFLAGP=0

NILNG=NILNG+1
IFLAGP=1

20 CONTINUE
50 CALL ECRO(AASC, ISP1+IECP=1, LP13, IDUM)
60 XTE=X(IPJ) YTE=Y(IPJ)
70 CALL PSUB1(IN)
80 IF(IN .EQ. NEOB) GO TO 100
90 X4=X(IPJP) Y4=Y(IPJP)
100 X4=X(IPJP) Y4=Y(IPJP)
110 X1=X(IPJ) Y1=Y(IPJ)
120 X2=X(IPJP) Y2=Y(IPJP)
130 X3=X(IPJP) Y3=Y(IPJP)
140 X4=X(IPJ) Y4=Y(IPJ)
150 X1=X(IPJ) Y1=Y(IPJ)
160 X2=X(IPJ) Y2=Y(IPJ)
170 X3=X(IPJP) Y3=Y(IPJP)
180 X4=X(IPJP) Y4=Y(IPJP)
190 X1=X(IPJ) Y1=Y(IPJ)
200 X2=X(IPJP) Y2=Y(IPJP)
210 X3=X(IPJP) Y3=Y(IPJP)
220 X4=X(IPJ) Y4=Y(IPJ)
230 X1=X(IPJ) Y1=Y(IPJ)
240 X2=X(IPJP) Y2=Y(IPJP)
250 X3=X(IPJP) Y3=Y(IPJP)
260 X4=X(IPJP) Y4=Y(IPJP)
270 X1=X(IPJ) Y1=Y(IPJ)
280 X2=X(IPJP) Y2=Y(IPJP)
290 X3=X(IPJP) Y3=Y(IPJP)
300 X4=X(IPJ) Y4=Y(IPJ)
310 X1=X(IPJ) Y1=Y(IPJ)
320 X2=X(IPJP) Y2=Y(IPJP)
330 X3=X(IPJP) Y3=Y(IPJP)
340 X4=X(IPJ) Y4=Y(IPJ)
350 X1=X(IPJ) Y1=Y(IPJ)
360 X2=X(IPJP) Y2=Y(IPJP)
370 X3=X(IPJP) Y3=Y(IPJP)
380 X4=X(IPJ) Y4=Y(IPJ)
390 X1=X(IPJ) Y1=Y(IPJ)
400 X2=X(IPJP) Y2=Y(IPJP)
410 X3=X(IPJP) Y3=Y(IPJP)
420 X4=X(IPJ) Y4=Y(IPJ)
430 X1=X(IPJ) Y1=Y(IPJ)
440 X2=X(IPJP) Y2=Y(IPJP)
450 X3=X(IPJP) Y3=Y(IPJP)
460 X4=X(IPJ) Y4=Y(IPJ)
470 X1=X(IPJ) Y1=Y(IPJ)
480 X2=X(IPJP) Y2=Y(IPJP)
490 X3=X(IPJP) Y3=Y(IPJP)
500 X4=X(IPJ) Y4=Y(IPJ)
510 X1=X(IPJ) Y1=Y(IPJ)
520 X2=X(IPJP) Y2=Y(IPJP)
530 X3=X(IPJP) Y3=Y(IPJP)
540 X4=X(IPJ) Y4=Y(IPJ)
550 X1=X(IPJ) Y1=Y(IPJ)
560 X2=X(IPJP) Y2=Y(IPJP)
570 X3=X(IPJP) Y3=Y(IPJP)
580 X4=X(IPJ) Y4=Y(IPJ)
590 X1=X(IPJ) Y1=Y(IPJ)
600 X2=X(IPJP) Y2=Y(IPJP)
610 X3=X(IPJP) Y3=Y(IPJP)
620 X4=X(IPJ) Y4=Y(IPJ)
630 X1=X(IPJ) Y1=Y(IPJ)
640 X2=X(IPJP) Y2=Y(IPJP)
650 X3=X(IPJP) Y3=Y(IPJP)
660 X4=X(IPJ) Y4=Y(IPJ)
670 X1=X(IPJ) Y1=Y(IPJ)
680 X2=X(IPJP) Y2=Y(IPJP)
690 X3=X(IPJP) Y3=Y(IPJP)
700 X4=X(IPJ) Y4=Y(IPJ)
710 X1=X(IPJ) Y1=Y(IPJ)
720 X2=X(IPJP) Y2=Y(IPJP)
730 X3=X(IPJP) Y3=Y(IPJP)
740 X4=X(IPJ) Y4=Y(IPJ)
750 X1=X(IPJ) Y1=Y(IPJ)
760 X2=X(IPJP) Y2=Y(IPJP)
770 X3=X(IPJP) Y3=Y(IPJP)
780 X4=X(IPJ) Y4=Y(IPJ)
790 X1=X(IPJ) Y1=Y(IPJ)
800 X2=X(IPJP) Y2=Y(IPJP)
810 X3=X(IPJP) Y3=Y(IPJP)
820 X4=X(IPJ) Y4=Y(IPJ)
830 X1=X(IPJ) Y1=Y(IPJ)
840 X2=X(IPJP) Y2=Y(IPJP)
850 X3=X(IPJP) Y3=Y(IPJP)
860 X4=X(IPJ) Y4=Y(IPJ)
870 X1=X(IPJ) Y1=Y(IPJ)
880 X2=X(IPJP) Y2=Y(IPJP)
890 X3=X(IPJP) Y3=Y(IPJP)
900 X4=X(IPJ) Y4=Y(IPJ)
910 X1=X(IPJ) Y1=Y(IPJ)
XLA = Y41 * X12 = Y12 * X41

IF (XNU, NE, 0) GO TO 80

THE = XLA / XNU

IF (X41, NE, 0) GO TO 90

ETA = Y12 / Y41

GO TO 190

80 THE = XNU * SORT (XNU * XNU = u, XMU = XLA) / (P * XMU)

90 ETA = (X12 * X21 * THE) / (X41 * X3421 * THE)

100 OME = ETA / THE

UK = OME + OMT * THE + OME + ETA * THE + ETA * THE + ETA * THE

VUK = OME + OMT * ETA + OME + ETA * THE + ETA * THE + ETA * THE

IF (MPTST, EN, 1) GO TO 110

110 UK = OME * OME + ETA * ETA

VUK = OME * VUK

UKS = UK

VKS = VK

XTE = XTE + DT * UK

YTE = YTE + DT * VUK

MPTST = 1

GO TO 60

110 UK = OME * OME + ETA * ETA

VUK = OME * VUK

UKS = UK

VKS = VK

XTE = XTE + DT * UK

YTE = YTE + DT * VUK

MPTST = 1

GO TO 60

C SEE IF TIME-DEPENDENT PARTICLE DATA IS TO BE COLLECTED THIS CYCLE

C IF (IFLAGP, EQ, 0) GO TO 120

C YES, SEE IF THE CURRENT PARTICLE IS TO BE SAVED.

C IF (MOD(NPPT, NIST), NE, 0) GO TO 120

C YES, SAVE IT.

C NN = NPPT / NIST

C IBEGIN = NLCP1 + NLCP2 + 2 * (NP1 * (NILNG = 1) + NN = 1)

C PAX (NN, NILNG) = XTE

C PAY (NN, NILNG) = YTE

C CALL ECWR (XTE, IBEGIN, 2, IDUM)

C CONTINUE

P8OMH = AMAX1 (YTE, P8OMH)

PRTE = AMAX1 (XTE, PRTE)

PTOP = AMAX1 (YTE, PTOP)

C IF (NPPT, EQ, NPT) GO TO 140

C KP = KP + 3

C IF (KP, LT, LPB) GO TO 40

C CALL ECWR (AASC, NLCP1 + IECP = 1, LPB, IDUM)

C IECP = IECP + LPB

GO TO 30

C CALL ECWR (AASC, NLCP1 + IECP = 1, LPB, IDUM)
IF (NCYC .EQ. 1) PTPOLD = PTP
RETURN
150 CONTINUE
IF (IFLAG .EQ. 2) GO TO 130
160 XX1 = X1
YY1 = Y1
XX2 = X2
YY2 = Y2
CALL PSUB2(IT, IFLAG)
IF (IFLAG .NE. 0) GO TO 150
IF (IT .EQ. 0) GO TO 170
IF (I .EQ. 1) GO TO 250
I = I + 1
GO TO 170
170 XX1 = X4
YY1 = Y4
XX2 = X3
YY2 = Y3
CALL PSUB2(IT, IFLAG)
IF (IFLAG .NE. 0) GO TO 150
IF (IT .EQ. 0) GO TO 180
IF (I .EQ. 1) GO TO 250
J = J + 1
GO TO 210
180 XX1 = X3
YY1 = Y3
XX2 = X2
YY2 = Y2
CALL PSUB2(IT, IFLAG)
IF (IFLAG .NE. 0) GO TO 150
IF (IT .EQ. 0) GO TO 190
IF (I .EQ. 1) GO TO 250
J = J + 1
GO TO 220
190 XX1 = X4
YY1 = Y4
XX2 = X1
YY2 = Y1
CALL PSUB2(IT, IFLAG)
IF (IFLAG .NE. 0) GO TO 150
IF (IT .EQ. 0) GO TO 200
IF (J .EQ. 2) GO TO 250
J = J + 1
EC = (J = 1) * NQI
190 IEC = (J = 1) * NQ + ISC2
JOLD = J
CALL ECPOD(AASC(ISC2), IEC, NQI2, IDUM)
GO TO 220
210 IX = (I = 1) * NQ + ISC2
IPJ = IJ + NQI
IJP = IJ + NQI
220 X1 = X(IPJ)
Y1 = Y(IPJ)
X2 = X(IPJP)
Y2 = Y(IPJP)
X3 = X(IPJP)
Y3 = Y(IPJP)
257       Xa=y(IJ)
258       Ya=y(IJ)
259       CALL PSUB1(IN)
260       IF(IN.EQ.0) GO TO 140
261       XTE=XTE
262       YTE=YTE
263       ITAB(NPPT)=(J-1)*IP1+I
264       GO TO 70
265      230 CONTINUE
266      DO 240 IPX=IPD,IPD+6
267      240 WRITE(IPX,280) NPPT, ITAB(NPPT), XTE, YTE, XEN, YEN, X1, Y1
268      ITAB(NPPT)=(J-1)*IP1+I
269      GO TO 130
270      250 XPAR(XP)=XP+6
271      DO 260 IPX=IPD,IPD+6
272      260 WRITE(IPX,270) NPPT, ITAB(NPPT)
273      GO TO 130
274      275 FORMAT(5X,BXPARTICLE,110,25HTOSSED OUT, CELL NUMBER,110)
276      280 FORMAT(5X,14HEPR PARTICLE/15,6F7.4)
277      END

********************************************************************
1 C     SUBROUTINE PSUB1(IN)
2 C    ORIGINALLY WRITTEN BY A.A.AMSDEN AND HANS RUPPEL, LASL T=3
3 C    MODIFIED BY J.L. NORTON, LASL T=3, 1975
4 C
5 C *  ----- BEGIN COMDECK YSTORE -----
6 C *  ----- BEGIN COMDECK YAGIDM -----
7 C DIMENSION X(1), XPAR(1), R(1), XPAR(1), Y(1), XPAR(1), U(1), V(1), DELSM(0)
8 C 1OTOR, ROT(1), SIE(1), MP(1), RPM(1), RSEQ(1), T(1), ETIL(1), RVOL
9 C 1 T(1), M(1), ANC(1), F(1), PL(1), UL(1), VR(1), VTIL(1)
10 C 3, VL(1), ROL(1), AVXS(1), AVYV(1), DLSRO(1), DOLSRO(1), CAPGAM(1), TUG
11 C 4, SIG(1), THS(1), GHRD(1), GDP(1), TURVD(1), TMLT(1)
12 C 5, CONC(1), CTEM(1), ANC(1), ANC(1), GRSV(1), GZSV(1), X13K(1), X24K(1)
13 C 6, Y13K(1), Y24K(1), XR13K(1), XR24K(1), OKLSM(1), AREA(1)
14 C *  ----- END COMDECK YAIDM -----
15 C *  ----- BEGIN COMDECK YAGSC -----
16 C LOGICAL HIST, FILM, PAPR, TURN
17 C REAL LAM, MU
18 C COMMON/YSC1/AASC(NSCP)
19 C COMMON/YSC2/AASC(9600)
20 C COMMON/YSC2/AA(1), ANC, A,FAC, A,B, AM, BOL, COLAM, CYL, DR, DT, DTC, DTFC,
21 C 1 DTC(10), DT0(10), DT08, DT06, DT04, DT02, DTV, EZ, EM, EPS, FIPX, FIPX,
22 C 2 FIPX, FIPX, FIPX, FIPX, FIPX, FIPX, FREZ4, GR, GROVEL, GZ, GZP, I, IBAR,
24 C 4 UINF, IXM, IYM, IYT, J, JBAR
25 C COMMON/YSC2/JCEN, JP1, JP2, JP4, JUNF, JUNF02, KXI, LAM, LPA, MU, NAME(8),
27 C 1 NCYC, NCL, NPS, NPT, NQ, NQI, NQ1B, NQ2, NSEC, NUMIT, ZP1G, Q1, QM, QMCL, PXCONV
28 C 2, PXL, PXR, PXY, PXY, PXY, PXY, PXY, PXY, HZROH, REZSD, REY2, RIB, RIBR,
29 C 3 FFH2ZT, FREZH, ROBH, T, THIRD, NCLST, TOUT, TFWIN
COMMON/YSC2/TUQI, TUSI, NCG, TNEG, TNEGTV, TUSV, TURB, PTOV, PRITX, PBOTH, 
1. ILNG, NILNG, TP3, TUPG, TDQSAV, TK, TI, TUDNGEP1, SASV1, QLEVEL, TG, IST, 
2. VV, XCONV, XL, XR, YB, YCONV, YT, PTPOLD, DTSV, DTLAST, FYBO, FYBO, VCNVL, 
3. XCNVL, FIXRO, FIXLO, IXRO, IXLO, ISVW, JBVW, QM, QMX, QMX, JNH, J2, TLM, 
4. ROMFX, ROMFY, ROMFB, JUOMP, TWHRO, TE, TMASS, DTVSAV, DTSVAV, IDTV 
5. JTV, JDT, JDT, JTC, JTC, TSC, POTE, UMOM, VMOM, THAX, TGMX, ITM, JTM, ITO, JTG 
6. TMASS, WMAXF, RMINEF, TRSTRD, 

COMMON/YSC2/ZZ 

COMMON/YSC2/ITAB(ITABP) 

COMMON/YSC2/ITAB(1000) 

COMMON/YSC2/PSTRT,FILM,PAPER,IPD,IPD 

**** END COMDECK YAOEQ ****

EQUIVALENCE(AASC(1),X,YPAR), (AASC(2),Y,IPAR), 

1 AASC(4),Y), (AASC(5),V), (AASC(6),RO), (AASC(7),DELSM,RSOM,MP), (AASC 

1 (A),E,ETIL,AREA,XR13K), 

2 (AASC(15),SIF), (AASC(16),PM0,DKLSM,RMP), (AASC(9 

3 ),RVOl), (AASC(18),M,H,RH,VP), (AASC(11),P,PL,EP,UP), (AASC(12),UTIL, 

4 U,L,PMK,PI1), (AASC(13),VTIL,VL,PMR,PR), (AASC(14),Q,CO,ROL), (AASC(17 

5 ),CAPKH,UC), (AASC(18),TUG), (AASC(19),SIG), (AASC(20),TO), (AASC( 

6 21),GROR), (AASC(22),GRORO), (AASC(23),I), (AASC(24),GZSV 

7 ), (AASC(25),NLSROG,VC), (AASC(26),GRSV), (AASC(27),GROR,TO), 

8 R24K), (AASC(28),M,MTI), (AASC(29),CONC), (AASC(30),CCTRL,XR24K), 

9 AASC(31),ANCX), (AASC(32),ANCX), (AASC(33),AXXY,S,X13K), (AASC(34), 

1 AVY2K,X24K 

REAL M,MP,MPAR,MTIL 

**** END COMDECK YAOEQ ****

**** BEGIN COMDECK PCOM ****

EQUIVALENCE(X(1),X),(Y(1),Y),(X(2),X2),(X(3),X3),(X(4),X4) 

EQUIVALENCE(Y(1),Y),(Y(2),Y1),(Y(3),Y3),(Y(4),Y4) 

**** BEGIN COMDECK PCOM ****

DIMENSION AMZ(7) 

DATA PI/3.1415926535897932384626/ 

DATA PI2/1.5707963267489896192313/ 

DATA PI2/6.2831853071798647649252/ 

10 AMZM=1,E=10 

DO 60 K=1,4 

IF(X(1(K),EQ,XTK) GO TO 40 

XX=(Y(1(K)-YTK))/X(1(K)-XTK) 

YY=ABS(XX) 

IF(YY,GT,1.0) GO TO 30 

SLOPE=PI2/XX 

GO TO 30 

20 SLOPE=SIGN(PI2,1./YY,XX) 

30 IF(XTFN,GX(1(K)) SLOPE=SLOPE+PI 

78 IF(XTK+LT,X(1(K)) AND,SLOPE+LT,0.) SLOPE=SLOPE+PI2 

GO TO 50 

80 SLOPE=SIGN(PI2,1.,YY,XX) 

5A AMZ(K)=SLOPE 

6A AMZM=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K) 

AMZ(K)=AMX,AMZ(K)
87 AMZ(7) = AMZ(3) = AMZMIN
88 AMZ(3) = AMZ(7)
89 AMZ(4) = AMZ(4) = AMZMIN
90 IF (AMZ(1) .EQ. 0.) K = 1
91 IF (AMZ(2) .EQ. 0.) K = 2
92 IF (AMZ(3) .EQ. 0.) K = 3
93 IF (AMZ(4) .EQ. 0.) K = 4
94 NTE = 0
95 IN = 0
96 IF (AMZ(K+3), GT, AMZ(K+2), AND, AMZ(K+2), GT, AMZ(K+1), AND, AMZ(K+1), GT, A
97 1 AMZ(K)) NTE = 1
98 IF (INTE, EQ, 0) RETURN
99 IN = 1
100 DO 100 K = 1, 4
101 AMZ1 = AMZ(K)
102 AMZ2 = AMZ(K+1)
103 AMZ3 = AMZ(K+2)
104 IF (AMZ1 .EQ. AMZ2, NE, 0.) GO TO 70
105 AMZ2 = AMZJ(AMZ1, AMZ2)
106 IF (AMZ3, LT, AMZ2) GO TO 80
107 70 PHI = ARG(AMZ1, AMZ2)
108 GO TO 90
109 80 PHI = PI*2 - AMZ
110 90 IF (PHI, EQ, PI) IN = 0
111 IF (IN, EQ, 0) RETURN
112 1 PD CONTINUE
113 RETURN
114 END

SUBROUTINE PSUB2(IT,IFLAG)

ORIGINALLY WRITTEN BY A. A. AMSDEN AND HANS RUPPEL, LASL T=3
MODIFIED BY J. L. NORTON, LASL T=5, 1975

* ----- BEGIN COMDECK YSTORE ----- *
* ----- BEGIN COMDECK YAQSC ----- *
* ----- END COMDECK Yyrıca ----- *
* ----- BEGIN COMDECK YAQSC ----- *
* ----- END COMDECK YAQSC ----- *

LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAM, MU
COMMON/YSC1/AAASC(NASC1)
COMMON/YSC1/AAASC(NASC0)
COMMON/YSC2/AAASC(NASC0)
COMMON/YSC2/AAASC(0)

1 DTO(10), DTO2(10), DTO3, DTO4, DTO5, DTO, DTV, DZ, EHM, EPS, FIPXL, FIPXR,
2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2 2s 2

COMMON/YSC2/ITAB(ITABP)
COMMON/YSC5/RESTRT,FILH,PAPER,IPD,IFD

* ***** END COMDECK YAGSC *****
* ***** BEGIN COMDECK YAGED *****

EQUIVALENCE(AASC(1),X,XPAR), (AASC(2),R,YPAR), (AASC(3),Y,HPAR),
1 AASC(4),U), (AASC(5),V), (AASC(6),RO), (AASC(7),DEL,RCSG,MP), (AASC
2 (8),F,ETIL,AREA,XY13K),
3 (AASC(15),SIF), (AASC(16),PM,DKLSM,RMP), (AASC0
4 (AASC(10),R,VP), (AASC(11),P,PL,EP,UP), (AASC(12),UTIL,
5 UL,PMX,PU), (AASC(13),VTIL,VL,PMY,VP), (AASC(14),G,CO,ROL), (AASC(17
6 ),TAPAGA,UG), (AASC(18),TU3), (AASC6),SIG), (AASC(20),TUS), (AASC
7 (AASC(21),GROR), (AASC(22),GRROZ), (AASC(23),DSLRG1,Y13K), (AASC(24),GZSV
8 ), (AASC(25),DLSROV,YG), (AASC(26),GRSV), (AASC(27),GRROP,TUGVEC,
9 (AASC(28),MITL), (AASC(29),CONC), (AASC30),CTEMP,XY24K),
1 (AASC31),ANCU), (AASC32),ANCV), (AASC33),AVXSV13K), (AASC34),
2 AVYSV,X20K)

REAL H,MP,MPAR,MTIL

* ***** END COMDECK YAGED *****
* ***** BEGIN COMDECK PCOM *****

COMMON/XTENC/XTEN,YTEN,XTE,YTE,X1(4),Y1(4),XX1,YY1,XX2,YY2

EQUIVALENCE(X1(2),X2), (X1(3),X3), (X1(4),X4)

EQUIVALENCE(Y1(2),Y2), (Y1(3),Y3), (Y1(4),Y4)

* ***** END COMDECK PCOM *****

DATA ALF/ALF2/0,081,1/08000001/

10 IDAG0

IF((YTEN=YY2)*(XTEN=XX1),EG,YY1)*(XTEN=XX2)) GO TO 30

IF((X1=XX2),LT,EM10) GO TO 50

PSEL(YY2,YTE)/(XTEN=XTE)

IF((ABS(X1)=XX2),LT,EM10) GO TO 50

GSL=YY1)/(XX2=XX1)

IF((PSL,EG,GSL)) RETURN

YINS=YY1*PSL+(XTEN=GSL*XX1)/(PSL=GSL)

YINS=YY1*GSL/(XX1=XX1)

20 ITAG=0

IF((XX1=XINS)*XX2=XINS,LE,0),AND,((YY1=YINS)*YY2=YINS,LE,0)

1 ) ITAG=1

IF((X1=XX1)*XINS,LE,0),AND,((YY1=YY2)*Y1=YINS,LE,0)

1 ) ITAG=1

IF((ITA,EG,2) GO TO 70

19
SUBROUTINE PTBDF(XJ,YJ)

C ROUTINE TO ADD TURBULENT DIFFUSION EFFECTS TO THE PARTICLE
MOTION

C (XJ,YJ) IS THE POSITION OF THE PARTICLE AFTER
THE EFFECTS OF CONVECTION HAVE BEEN ADDED, THE POSITION
WILL BE RETURNED WITH THE TURBULENT DIFFUSION CORRECTION
ADDED,

C WRITTEN BY J.L. NORTON, LASL T=3, 1975

C ----- BEGIN COMDECK YSTORE ----- 
C ----- BEGIN COMDECK YADIM ----- 

DIMENSION X(1), XPAR(1), XR(1), XPAR(1), Y(1), XPAR(1), YPAR(1), U(1), UG(1), DELSM(1), V(1), VG(1), RO(1), SIF(1), MP(1), RHP(1), RCS(1), E(1), ETL(1), RVOL(1), M(1), RM(1), VP(1), P(1), PL(1), VP(1), UTL(1), UL(1), CO(1), UTIL(1)

RETURN
30 XTN=ALF2*XTEN+(1.-ALF2)*XTE*ALF
40 YTN=ALF2*YTEN+(1.-ALF2)*YTE
50 XPAR(KP)=XTEN
60 XPAR(KP)=YTEN
70 IFLAG=1
80 DO 40 IPX=IPD,IFD,6
90 RETURN
100 FORMAT(5X,7HIFLAG=##)
110 END

RETURN
30 XTN=ALF2*XTEN+(1.-ALF2)*XTE*ALF
40 YTN=ALF2*YTEN+(1.-ALF2)*YTE
50 XPAR(KP)=XTEN
60 XPAR(KP)=YTEN
70 IFLAG=1
80 DO 40 IPX=IPD,IFD,6
90 RETURN
100 FORMAT(5X,7HIFLAG=##)
110 END
18 3,VL(1),ROL(1),AVXSV(1),AVYSV(1),DLSROQ(1),DLSROQ(1),CAPGAM(1),TUQ
19 4,(1),SIG(1),TUS(1),GRRO1(1),GRRO2(1),GRRO3(1),GRRO4(1),TUVEC(1),MTIL(1),
20 5,CONC(1),CTEMP(1),ANCV(1),ANCY(1),GRSV(1),GZSV(1),X13K(1),X24K(1),
21 6,Y15K(1),Y24K(1),X13K(1),X24K(1),DIIKSV(1),AREA(1)
22 * ---- END COMDECK YAOQIM ----
23 * ---- BEGIN COMDECK YAOQSC ----
24 LOGICAL RESTRT, FILM, PAPER, TURB
25 REAL LAM, MU
26 COMMON/YSCI/AASC(NSCP1)
27 COMMON/YSCI/AASC(96RO)
28 COMMON/YSCI/AASC(1),ANC, AO, A0FAC, AOH, BB, COLAMU, CYL, DR, DT, DTC, DTFAC,
29 1 DT(I0), DT02, DT08, DTPO, DTZ, EM10, EPS, FIPXL, FIPXR,
30 2 FIPYB, FIPYT, FIXL, FIXR, FIYB, FIYT, FREXZK, GR, GROVEL, GZ, GZP, I, IBAR,
31 3 IDTO, IJ, IJM, IM1, IPXL, IPXR, IPYB, IPYT, IP1, IP2, ISC2, ISC3, IV,
32 4 IUNF, IXL, IXR, IYB, IYT, IYBAR
33 COMMON/YSCI/JCEN, JP1, JP2, JPU4, JUNF, JUNFO2, KV1, LAM, LBP, MU, NAME(8),
34 1 NCV, NLC, NPS, NPT, NZQ, NZ1B, NZ12, NZ2, NUM, NUTIL, ON, OMCYL, PCONV
35 2, PXL, PXR, PYB, PYCONV, PYT, ROT, REZRO, REZSIE, REZYO, REZOR, RIBAR, RIBJB,
36 3 FREXZ, FREZB, ROME, TTHRD, TOUT, TWF
37 COMMON/YSCI/TUGI, TUSI, NCO, TENG, TNEG, TUSV, TURB, TPTP, RITE, PBT, BTM,
38 1 IL, INLG, NLP, NPT, TQ, TI, TUGENG, EP1, SAV1, GLEVEL, TQ, IST,
39 2 VV, XCONV, XL, XR, YB, YCONV, YT, PTPO, DTV, DTV, DTV, DTV, DTV, DTV,
40 3 XCNVLN, FIXPO, FIXLO, IXRO, IXLO, IYSL, JBVN, QMN, QMX, WMA, JNM, T2, TLM,
41 4 ROMX, ROMY, ROMF, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ, TQ
42 5, JDV, JDT, JDC, JTC, JTI, JSP, JEC, JUMO, JUMO, JUMO, JUMO, JUMO, JUMO, JUMO,
43 6, THASS, XMAX, RMINEF, TSTRTD
44 COMMON/YSCI/ZZ
45 COMMON/YSCI/TAB(1),TAB(1)
46 COMMON/YSCI/ITAB(1),ITAB(1)
47 COMMON/YSCI/RESTRT, FILM, PAPER, IPD, IFD
48 * ---- END COMDECK YAOQSC ----
49 * ---- BEGIN COMDECK YAOQ ----
50 EQUIVALENCE(AASC(1),X,XPAR), (AASC(2),R,YPAR), (AASC(3),Y,MPAR),
51 1 AASC(4),U), (AASC(5),V), (AASC(6),RD), (AASC(7),DELSM,RCSG,MP), (AASC
52 2, R);, ETIL, AREA, X13K,
53 3, AASC(15),SIE), (AASC(16),PMO,DKLSR,MPMO), (AASC(I9)
54 4, RVO), (AASC(18),H,MP,VP), (AASC(11),P,PL,EP,UP), (AASC(12),ULTIL,
55 5, OMPX,PCO, (AASC(6),J),V), (AASC(11),PM,PM,PM,PM,PM), (AASC(19),G,CO,ROL),
56 6, CAPGAM,BAU), (AASC(15),THD), (AASC(19),G), (AASC(29),VUS), (AASC(31)
57 7, 21), GROVR), (AASC(22),GROZ), (AASC(23),DLSROG,Gr13K), (AASC(24),GZSV
58 8, ) (AASC(25),DLSQ,Gl3K), (AASC(26),GTVS), (AASC(27),GRO, TUVEC,
59 9, Y24K), (AASC(28),MTIL), (AASC(29),CONC), (AASC(30),CTEMP, X24K)
60 10, AASC(31),ANCY), (AASC(32),ANCY), (AASC(33),AVXSV, X13K), (AASC(34),
61 1 Avxsv, X24K
62 REAL MP, MPAR, MTIL
63 * ---- END COMDECK YAOQ ----
64 * ---- END COMDECK YSTORE ----
65 * ---- BEGIN COMDECK YBDIF ----
66 COMMON/CTDIF/FRFV(21), NERFV, DXEF
67 * ---- END COMDECK YBDIF ----
68 C IPASS=1 IS FOR X (R)
69 C IPASS=2 IS FOR Y (Z)
70 C
71 C
72 C
73 C
74 C
75 C
IF(XJ,LE, RMINEF) GO TO 50
GET A RANDOM NO. FROM A UNIFORM DISTRIBUTION
ON THE INTERVAL (0,1)
10 CONTINUE
XINPUT=0.
ETA1=RNUMF(XINPUT)
MAP EACH HALF OF THE INTERVAL (0,1) INTO (0,1)
ETA=ABS(1.-2.*ETA1)
FIND THE ERROR FUNCTION BOUNDS ON THE RANDOM NO.
DO 20 L=2, NERFV
20 IF(ETA,LE,ERFV(L)) GO TO 50
IF NO BOUNDS ARE FOUND, THE RANDOM NO. IS TOO CLOSE TO 1,
GO BACK AND TRY ANOTHER ONE.
GO TO 10
30 CONTINUE
BOUNDS HAVE BEEN FOUND, INTERPOLATE THE ERROR FUNCTION INVERSE,
ERFINV=DXEF*FLOAT(L-2)+(ETA-ERFV(L))/ERFV(L-1))
CALCULATE THE CORRECTION
CORR=ERFINV*SQRT(DT*SIG(IJ))
ADD ON THE SIGN OF THE CORRECTION
IF(ETA,GT,.5) CORR=CORR
GO CORRECT THE PROPER COORDINATE
GO TO (40,60), IPASS
X (R) COORDINATE
40 CONTINUE
XJ=XJ+CORR
50 CONTINUE
GO BACK AND DO THE Y (Z) COORDINATE
IPASS=2
GO TO 10
60 CONTINUE
Y (Z) COORDINATE
VJ=VJ+CORR
RETURN
SUBROUTINE REZONE

ROUTINE TO CALCULATE THE GRID VELOCITIES, NEW GRID POSITIONS, AND RESTORE THE AMBIENT ATMOSPHERE TO HYDROSTATIC EQUILIBRIUM

ORIGINALLY WRITTEN BY A.A.,AMSDEn AND HANS RUPPEL, LASL T-3
MODIFIED BY J.L. NORTON, LASL T-3, 1975

----- BEGIN COMDECK PARAM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ

----- BEGIN COMDECK YSTORE ----- COMMON/PCGM/NSCP1, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,

----- BEGIN COMDECK YADIM ----- DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UC(1), DLSM(1)

----- BEGIN COMDECK PARAM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,

----- BEGIN COMDECK YADIM ----- DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UC(1), DLSM(1)

----- BEGIN COMDECK YADIM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,

----- BEGIN COMDECK YADIM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,

----- BEGIN COMDECK YADIM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,

----- BEGIN COMDECK YADIM ----- COMMON/PDCM/NSCP, ITABP, ITABXP, ITABYP, IPF, NP1, NP2, NLCP1, NLCP2,
112 \quad \text{IF}(I,EQ,2) \quad \text{VG}(IMJ)=\text{VG}(IJ)
113 \quad \text{GO TO 130}
114
115 \quad 50 \text{ CONTINUE}
116 \quad \text{IF}(I,NE,1) \quad \text{GO TO 60}
117 \quad \text{UG}(IJ)=0,0
118 \quad \text{GO TO 70}
119
120 \quad 60 \quad \text{IF}(I,NE,IP1) \quad \text{GO TO 70}
121 \quad \text{UG}(IJ)=FCX
122 \quad \text{VG}(IJ)=\text{VG}(IMJ)
123 \quad 70 \quad \text{IF}(J,NE,2) \quad \text{GO TO 80}
124 \quad \text{UG}(IJ)=FC3
125 \quad \text{GO TO 90}
126
127 \quad 80 \quad \text{IF}(J,NE,JP2) \quad \text{GO TO 90}
128 \quad \text{UG}(IJ)=UG(IJM)
129 \quad \text{VG}(IJ)=FCF2
130 \quad 90 \quad \text{IF}(I,NE,1,OR,J,NE,2) \quad \text{GO TO 100}
131 \quad \text{VG}(IJ)=FC3
132 \quad 100 \quad \text{IF}(I,NE,IP1,OR,J,NE,2) \quad \text{GO TO 110}
133 \quad \text{UG}(IJ)=FCX
134 \quad \text{VG}(IJ)=FCF2
135 \quad 110 \quad \text{IF}(I,NE,1,OR,J,NE,JP2) \quad \text{GO TO 120}
136 \quad \text{UG}(IJ)=FC3
137 \quad 120 \quad \text{IF}(I,NE,IP1,OR,J,NE,JP2) \quad \text{GO TO 130}
138 \quad \text{UG}(IJ)=FCX
139 \quad \text{VG}(IJ)=FCF2
140 \quad 130 \quad \text{CONTINUE}
141 \quad \text{IJ}=IPJ
142 \quad \text{IJP}=IJP+NQ
143 \quad \text{IJM}=IJM+NQ
144 \quad \text{CALL LOOP}
145 \quad \text{CONTINUE}
146 \quad \text{IJ}=IJP
147 \quad \text{IJP}=IJP+NQ
148 \quad \text{IJM}=IJM+NQ
149 \quad \text{CALL LOOP}
150 \quad \text{CONTINUE}
151 \quad \text{Y}(IJ)=\text{Y}(IJ)+\text{VG}(IJ)\times DT
152 \quad \text{R}(IJ)=\text{X}(IJ)\times CYL+OMCYL
153 \quad \text{CONTINUE}
154 \quad \text{IJ}=IJ+NQ
155 \quad \text{CALL LOOP}
156 \quad \text{CONTINUE}
157 \quad \text{CALL DONE}
158 \quad \text{CALL FILMCO}
159 \quad \text{CALL START}
160 \quad \text{ETMP}=\text{REZIE}
161 \quad \text{ROTMP}=\text{REZRON}
162 \quad \text{CONC}=0$
163 \quad \text{CALL AIR}
164 \quad \text{XX}=\text{GMON+REZIE}
165 \quad \text{YY}=\text{ABS(GZ)}/\text{XX}
166 \quad \text{DO 200 J=2,JP1}
167 \quad \text{DO 190 I=1,IBAR}
168 \quad \text{IPJ}=IJ+NQ
SUBROUTINE RTAPE

ROUTINE TO READ IN ONE DUMP OF A YAGUI DUMP FILE (ALL BUT THE HEADER RECORD)

WRITTEN BY J.L. NORTON, LASL T=3, 1975

* ------ BEGIN COMDECK PARAM ------ *
COMMON/COM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFILMSZ
* ------ END COMDECK PARAM ------ *

* ------ BEGIN COMDECK YAGSC ------ *
LOGICAL RESTRT, FILM, PAPER
REAL LAM, MU

COMMON/YSC1/ASCI(NSCP1)
COMMON/YBC1/ASCI(9600)
COMMON/YSC2/AA(1), ANC, A0, AQAFAC, AGM, B0, COLAMU, CYL, DR, DT, DTC, DTFAC,
1 DTO(10), DTOC(10), DTO2, DTO8, DTPOS, DTV, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPYB, FIPYI, FIPYL, FIPX, FIYI, FIYT, FREZXR, GR, GRDVEL, GZ, GPZ, I, IBAR,
3 IDTO, IJ, IJM, IJP, IM1, IPXL, IPXR, IYP, IYP1, IP1, IP2, ISC2, ISC3, ITV,
4 IUNF, IXL, IXR, IYB, IYT, J, JBAR
COMMON/YSC2/JSCT, JP1, JP2, JPG, JUNF, JUNFQ2, KXI, LAM, LPB, MU, NAME(8),
1 NCYC, NLCP, NSC, LCP, NO, NO1, NO2, NSC, NUMIT, ZORIG, OM, OCYLO, PCONV
2 PXL, PXR, PYB, PYCONV, PYT, RDT, REZRON, REZSIE, REZYB, RIBAR, RIBJB,
3 FREZYN, FREZYNB, ROMPR, T, THORD, NCLST, TOUT, TMFIN
COMMON/YSC2/TUGI, TUSI, NCQ, TNEG, TNEG4, TUSY, TURB, TPOP, TPR, TPR4, TPR7,
1 TILN, NILNS, TP3, TURPT, TDQSAV, TK, TI, TUDENG, EP1, SAV1, QLEVEL, TQ, IST,
2 VV, XCONV, XL, XR, Y, YCONV, YT, TPTOLD, DTSV, DTLAST, FY80, FY80, WCNNL,
3 XCNVLO, FIXRO, FIXLO, IXRO, IXL0, ISYM, JSYM, GMN, QMN, MMAX, JMN, T2, T3,
4 ROMFXR, ROMFY, ROMFYB, JDUMP, TMTHRD, TE, DTR, TMASS, DTSAV, DTSCAV, IDTV
5 JDTV, JDT, JDT, JDC, CIRC, TIS, POTE, UMOD, VMOD, TMAX, TGMX, ITM, JTM, ITG, JTG
6 THASSV, WHASE, WIMN, WIMN, TSTRTO
COMMON/YSC2/IZ
COMMON/YSC4/ITABP(ITABP)
COMMON/YSC4/ITAB(10000)
COMMON/YSC5/RESTRT, FILM, PAPER, IPD, IPD
* ------ END COMDECK YAGSC ------ *
SUBROUTINE RIROW

ROUTINE TO READ ROW J FROM LCM TO SCM BUFFER ONE

ORIGINALLY WRITTEN BY A.A.AMSDEN, LASL T-3
MODIFIED AND DOCUMENTED BY J.L.NORTON, LASL T-3, 1974

* ----- BEGIN COMMON PARAM ----- COMMON/PSCM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLC1P1,NLC1P2,
1 NLC3P3,NLC4P4,IFLMSZ
11 * ----- END COMMON PARAM ----- ----- BEGIN COMMON YASC -----
13 LOGICAL RE8TRT,FILM,PAPER,TURB
14 REAL LAH,MU
15 C COMMON/YSCI/AASC(NSCP1)
SUBROUTINE SETIJ

ROUTINE TO SET THE SCM POINTER TO THE ITH ELEMENT IN ROW J,
ASSUMING ROW J IS IN SCM BUFFER ONE

ORIGINALLY WRITTEN BY A. A. AMSEN, LASL T-3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-3, 1974

----- BEGIN COMDECK PARAM -----
COMMON/PCOM/NSCP1,ITABP,ITABXP,IPFR,IP1,IP2,NLCP1,NLCP2,
1 NLCP3,NLCP4,IFLMSZ
----- END COMDECK PARAM -----

----- BEGIN COMDECK YAGSC -----
LOGICAL RESTRT,FILM,PAPER,TURB
REAL LAM,MU
COMMON/YSC1/AASC(NSCP1):
COMMON/YSC1/AASC(9600)

COMMON/YSC2/AA(1),ANC,AB,ABFAC,ABM,B0,COLAMU,CYL,DR,DT,DTDC,DTFAC,
1 DTO(18),DTDC(18),DTDO,DTPOS,DTV,DZ,EM10,EPS,FIPXL,FIPXR,
2 FIPYB,FIPYF,FIXL,FXR,FIYB,FIYT,FREZXR,GR,GROVEL,GZ,GZIP,I,IBAR,
3 IDTO,IJ,JP,IM,IPXL,IPXR,IPYB,IPYT,IP1,IP2,ISC2,ISC3,ITV,
4 IUNF,IXL,IXR,ITY,IZ,IBAR
COMMON/YSC2/JCEN,JP1,JP2,JP4,JUNF,JUNFO2,XXI,LAM,LPB,MU,NAME(8),
SUBROUTINE START

ROUTE TO TRANSFER DATA BETWEEN LARGE CORE AND SMALL CORE IN
ORDER TO MINIMIZE SMALL CORE REQUIREMENTS

ORIGINALLY WRITTEN BY AMSDEN, LAKL T=3
MODIFIED AND DOCUMENTED BY NORTON, LAKL T=3, 1974

--- BEGIN COMDECK PARAM ------
COMMON/PSC/PSCP1, ITABP, ITABX, ITABY, IPFB, NP1, NP2, NLCP1, NLCP2,
--- END COMDECK PARAM ------

LOGICAL RESTRT, FILM, PAPER, TURB

REAL LAM, MU

COMMON/YSC1/ASC(NSCP1)
COMMON/YSC1/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMMON/YSC2/ASC(9600)
COMMON/YSC2/ASC(0)
COMM...
6, TMASSV, XMAXEF, RMINEF, TSTRTO
COMMON/YSC2/II
C COMMON/YSC4/ITAB(1000)
COMMON/YSC5/RESTRT, FILM, PAPER, IPD, IFD
* ---- END COMDECK YAGSC ----
DATA NE/0/
C
READ IN THE FIRST THREE ROWS
C
IJPS IS THE SCM POINTER TO WHERE THE NEXT ROW IS TO BE READ
C INTO SCM, ONE NEEDS TO THINK OF THE SMALL CORE AREA AS BEING
C DIVIDED INTO THREE BUFFERS, SAY A, B, AND C, INITIALLY, ROW J=1
C IS READ INTO BUFFER A, ROW J=2 INTO BUFFER B, AND ROW J=3 INTO
C BUFFER C, WHEN A REQUEST IS MADE FOR ROW J=4, BUFFER A WHICH
C CURRENTLY HOLDS ROW J=1 IS WRITTEN OUT TO LCM AND ROW J=4 IS
C READ INTO BUFFER A. THIS PROCESS CONTINUES WITH ONE ROW BEING
C WRITTEN TO LCM AND ONE ROW REPLACING IT IN SCM SO THAT THREE
C ROWS ARE ALWAYS AVAILABLE, ONE IN EACH BUFFER.
C IECR IS THE LCM ADDRESS FROM WHICH DATA WILL BE READ TO SCM NEXT
C IFCW IS THE LCM ADDRESS TO WHICH DATA WILL BE WRITTEN FROM SCM NEXT
C
IJPS=1
IECR=0
IECW=0
C
READ THE FIRST ROW INTO SCM INTO BUFFER ONE
C AASC IS BEGINNING OF THE THREE SCM BUFFERS
C NQI IS THE NUMBER OF WORDS COMPRISING ONE ROW OF DATA
C NE IS AN ERROR FLAG WHICH IS UNUSED ON THE 7600
C
CALL ECRD(AASC(IJPS), IECR, NQI, NE)
C INCREMENT THE LCM READ ADDRESS
C IECR=IECR+NQI
C
SET THE BUFFER POINTER TO THE SECOND BUFFER, OBVIOUSLY,
C ISC2=NQI+1
IJPS=ISC2
C
READ IN THE SECOND ROW
C CALL ECRD(AASC(IJPS), IECR, NQI, NE)
C INCREMENT THE LCM READ ADDRESS AGAIN
C IECR=IECR+NQI
C
UPDATE THE POINTERS TO READ THE NEXT ROW INTO BUFFER THREE,
C IJP IS THE POINTER TO THE ELEMENT (1, J=1)
C IJ IS THE POINTER TO THE ELEMENT (1, J)
C IJM IS THE POINTER TO (1, J=1)
C IJMS IS THE SCM ADDRESS TO BE WRITTEN OUT NEXT
IBUF IS THE NEXT BUFFER TO BE USED WHEN THE SUBROUTINE IS ENTERED

10 CONTINUE
IJP=ISC3
IJPS=ISC3
IJ=ISC2
IJM=1
IJMS=1
IBUF=1

FILL THE THIRD BUFFER
20 CALL ECRD(AASC(IJPS),IECR,NQI,NE)

UPDATE THE LCM POINTER FOR THE NEXT READ
30 CONTINUE
IJP=1
IJPS=1
IJ=ISC3
IJM=ISC2

UPDATE THE LCM POINTER FOR THE NEXT WRITE

PREPARE THE POINTERS FOR THE NEXT READ DEPENDING ON WHICH BUFFER IS TO BE FILLED
40 CONTINUE
IJP=ISC2
IJS=ISC2
IJ=1
IJM=ISC3
IJMS=ISC3
C
INDICATE THAT BUFFER THREE IS TO BE FILLED NEXT TIME
C
IBUF=3
C
GO FILL BUFFER TWO
C
GO TO 20
C
ENTRY DONE
C
LOOP IS COMPLETED, WRITE OUT CONTENTS OF THE LAST TWO BUFFERS
C
THAT WERE LOADED, WRITE OUT BUFFER THAT WOULD HAVE BEEN
C
FILLED NEXT,
C
CALL ECWR(AASC(IJMS),IECW,NQI,IDUM)
C
IECW=IECW+NQI
C
DETERMINE WHICH IS THE OTHER BUFFER TO BE EMPTIED
C
GO TO (50,60,70),IBUF
C
BUFFER ONE HAS BEEN WRITTEN OUT, WRITE OUT BUFFER TWO,
C
50 IJMS=ISC2
C
GO TO 80
C
BUFFER TWO HAS BEEN WRITTEN OUT, WRITE OUT BUFFER THREE,
C
60 IJMS=ISC3
C
GO TO 80
C
BUFFER THREE HAS BEEN WRITTEN OUT, WRITE OUT BUFFER ONE,
C
70 IJMS=1
C
80 CALL ECWR(AASC(IJMS),IECW,NQI,IDUM)
C
RETURN
C
END

SUBROUTINE STARTD
C
THIS ROUTINE (AND ITS ENTRY POINT LOOPD) ALLOW ONE TO LOOP
C
BACKWARDS THROUGH THE MESH USING TWO SCM BUFFERS, ROWS J
C
AND J-1 ARE MADE AVAILABLE, THE CALL TO STARTD READS ROW JP2
C
INTO BUFFER TWO AND ROW JP1 INTO BUFFER ONE, AFTER STARTD,
IJMSEISC2

SET INITIAL SCM READ ADDRESS (BUFFER TWO)

SET NEXT LCM READ AND WRITE ADDRESSES,

ITV IS JP1*NQI OR THE LCM ADDRESS OF THE BEGINNING OF ROW JP2

IECR=ITV

IEC=ITV

READ ROW JP2 INTO BUFFER TWO

CALL ECRD(AASC(IJMS),IECR,NQI,IDUM)

DECREMENT THE READ ADDRESS

IECR=IECR-NQI
64 C PREPARE TO READ IN ROW J=1 INTO BUFFER ONE
65 C IJM POINTS TO (IP1,J=1) (THE END OF BUFFER ONE)
66 C
67 C 10 CONTINUE
68 C IJM=ISC2=NG
69 C IJMS=1
70 C IRUF=1
71 C IJM=IJ+NGI (THE LAST CELL OF BUFFER TWO)
72 C IJPS=ISC2
73 C CHECK TO SEE IF THE BOTTOM OF THE MESH HAS BEEN REACHED
74 C 20 IF(IECR,LT,0) GO TO 40
75 C NO, GO AHEAD AND READ IN THE NEXT BUFFER
76 C CALL ECRD(AASC(IJMS),IECR,NGI,IDUM)
77 C DECREMENT THE READ ADDRESS
78 C IECR=IECR-NGI
79 C 30 RETURN
80 C BOTTOM OF THE MESH HAS BEEN REACHED, SET A FLAG (IBUF=3), GO WRITE OUT ROW J=2, AND QUIT
81 C 40 IBUF=3
82 C GO TO 30
83 C
84 C ************************************************************
85 C ENTRY LOOP
86 C ************************************************************
87 C ENTRY TO CYCLE THROUGH ONE MORE ROW OF THE MESH IN A DESCENDING FASHION AS DESCRIBED ABOVE
88 C WRITE OUT ROW J AND READ ROW J=2 INTO ITS BUFFER
89 C
90 C 50 CONTINUE
91 C CALL ECWR(AASC(IJPS),IECWR,NGI,IDUM)
92 C DECREMENT THE WRITE ADDRESS
93 C IECWR=IECWR-NGI
94 C SET UP THE POINTERS FOR THE NEXT READ DEPENDING ON WHICH BUFFER IS TO BE FILLED, IF IBUF=3, ONLY A RETURN IS EXECUTED SINCE THE MESH IS FINISHED.
95 C GO TO (60,10,30),IBUF
96 C
SUBROUTINE TICBOX

ROUTINE TO DRAW A BOX AROUND THE GRID AREA DISPLAYED
AND LABEL THE AXES

ORIGINALLY WRITTEN BY A. A. AMSDEN, LASL T-3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-3, 1975

BEGIN

COMMON/Pavn/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCPL, NLCPL2,
1 NLCPL3, NLCPL4, IFMSZ

* ----- BEGIN COMDECK PARAM ----- *
1 COMMON/Pcom/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCPL, NLCPL2,
1 NLCPL3, NLCPL4, IFMSZ

LOGICAL RESTRT, FILM, PAPER, TURB
REAL LAH, MU

COMMON/YSC1/ASCII(NSCP1)
COMMON/YSC1/ASCII(9600)
COMMON/YSC2/ASCII(1), ASC, A0, A0FAC, A0M, B0, COLANU, CVL, DR, DT, DTC, DTFAC,
1 DTC(10), DTC(10), DTC(10), DTC(O), DTC(Op), DTC(D), DTV, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPYB, FIPYT, FIXL, FIXR, FIYB, FIYT, FREEZXR, GR, GRDVEL, GZ, GZP, I, IBAR,
3 IDTO, IJ, IM, IML, IPXL, IPXR, IPYB, IPYT, IP1, IP2, IISC2, IISC3, ITV,
4 UNF, IXL, IXR, IYB, IYT, IJBAR
5 COMMON/YSC2/JCEN, JP1, JP2, JP4, JUNF, JUNFO2, KXY, LAH, LPB, MU, NAME(8),
2 NCYC, NLP, NPS, NP1, NP2, NQ1, NQ2, NQC, NUMIT, ZORIG, ON, OM, OMCYL, PXCNGV
3 , PX, PXR, PYB, PYCNGV, PYT, RDT, REZRON, REZSI, REZY, RIBAR, RIBJB,
4 FREZYT, FREZYB, ROMEF, T, THIRD, NCLST, TOUT, TWF

COMMON/YSC2/TUDI, TUSI, NCSN, TSEG, TNGTSV, TUSV, TURB, PTOP, PRIE, POTH,
1 ILNG, NILNG, TP3, TUPOT, TDGSAV, TK, TI, TUGENG, EP1, SAV1, SAV2, QLEVEL, TO, IST,
1 VV, XCONV, XL, XR, YB, YCONV, YT, TPTOLD, DTSV, DLAST, FIYB0, IFYBO, YCNVL0,
3 XCNVL0, FIXRO, FIXL0, FIXL0, FIXL0, ISVW, JSVW, QMV, QMX, WMX, JNH, T2, HLIM,
4 ROMFXR, ROMFYB, JUMP, TMTHRD, TE, DTR, TMASS, DTSAV, DTCNV, IDTV
5 , JDTV, IDTC, JDTF, CIRC, TIS, POTE, UMOM, VMOM, TMX, TMX, ITH, JTM, ITG,
3 TMASSV, WMXEF, RMEIF, FMXEF, FSTRTD

COMMON/YSC2/ZZ
COMMON/YSC4/ITAB(ITABP)
COMMON/YSC4/ITAB(1000)
COMMON/YS5/RESTRT,FILM,PAPER,IPD,IFD

* END COMDECK YAGSC
* BEGIN COMDECK PCALL
* END COMDECK PCALL
* COMMON/XCONVP,YCONVP,YLP,ILP
DATA BCD/1H /

DRAW THE PLOT FRAME

CALL DRV(IYL,IYT,IXL,IXR)
CALL DRV(IXR,IYT,IXL,IXL)
CALL DRV(IXL,IXL,IYL,IYL)

IX2,IX3,IY2, AND IY3 ARE THE RASTER COORDINATES OF THE ENDS OF THE TIC MARKS TO BE DRAWN ON THE PLOTTING RECTANGLE

Ix2=IxL+8
IX3=IXR-8
IY2=IYB-8
IY3=IYT+8

ESTABLISH TOP AND BOTTOM FOR PLOT INCREMENTS, ROUND TOP AND BOTTOM GRID VALUES TO THE NEAREST POWER OF 10, (YUPI,YLBI),

IXX=ALOG10(YUP)
DTIC=10*XIX

DETERMINE THE Y EXTREMA TO USE TO SET DTIC, THE TIC INCREMENT.

THE Y MINIMUM, YLBI, IS SET SUCH THAT IT IS LESS THAN YLB AND EQUAL TO AN INTEGRAL MULTIPLE OF THE POWER OF TEN NEAREST TO BUT LESS THAN YUP; THE Y MAXIMUM, YUPI, IS DETERMINED SIMILARLY BUT SO THAT IT IS GREATER THAN YUP.

ASSUME THAT YLB IS .GE.0

YLBI=0,
IF(DTIC,GE,YLB) GO TO 20
DO 10 I=1,9
YLBI=YLBI+DTIC
IF(YLBI,LT,YLB) GO TO 10
YLBI=YLBI+DTIC
GO TO 20

CONTINUE
CALL UNCLE(1,6HTICBOX,24,24HCOULD NOT DETERMINE YLBI)

CONTINUE
YUP=DTIC
DO 30 I=2,10
YUP=YUP+DTIC
30 IF(YUPI,GE,YUP) GO TO 40

CONTINUE
CALL UNCLE(1,6HTICBOX,24,24HCOULD NOT DETERMINE YUPI)

CONTINUE

BOUNDS ESTABLISHED, DETERMINE THE LABELLING INCREMENT BASED ON 10 INCREMENTS.
DTIC=(YUPI-YLBI)*1
50 CONTINUE

COMPUTE HOW MANY TICS WILL BE DRAWN

ITIC=0
YTICT=YLBI
60 CONTINUE
IF(YTICT.GE.YLBI) GO TO 70
YTICT=YTICT+DTIC
GO TO 60
70 CONTINUE
YLBI=YTICT+DTIC
80 CONTINUE
IF(YTICT.GE.YUP) GO TO 90
ITIC=ITIC+1
YTICT=YTICT+DTIC
GO TO 80
90 CONTINUE

MUST BE AT LEAST 6 TICS OR DTIC WILL BE REDUCED

IF(ITIC.GE.6) GO TO 100
DTIC=DTIC/2
GO TO 50
100 CONTINUE

YTIC IS THE Y-COORDINATE OF THE LABEL

YTIC=YLBI
110 CONTINUE
IF(YTICT.GE.YLB) GO TO 120
YTICT=YTICT+DTIC
GO TO 110
120 CONTINUE

COMPUTE RASTER COORDINATE FOR YTIC

120 CONTINUE
IYIFFIYB+(YTIC-YLB)*YCONVP
130 CONTINUE
IF(IYI,LT,IYT) GO TO 210
DRAW TICS ON THE LEFT AND RIGHT SIDES OF THE FRAME
CALL DRV(IXL,IY1,IX2,IY1)
CALL DRV(IX3,IY1,IXR,IY1)
PREPARE AND OUTPUT THE LABEL ITSELF
IMSIGN=1H
IF(YTIC,GT,0) GO TO 140
IF(YTIC,LT,0) GO TO 130
IMSIGN=2H0.
CALL NCODE(8, 6H(6XA2), IMSIGN, 1, BCD)
GO TO 200
CONTINUE
XLOG=ALOG10(ABS(YTIC))
IMSIGN=1H=
GO TO 150
CONTINUE
XLOG=ALOG10(YTIC)
CONTINUE
IEXP=XLOG
XMAN=XLOG=FLOAT(IEXP)
IF(XMAN,GE,0.) GO TO 160
XMAN=XMAN+1.
IEXP=IEXP+1
CONTINUE
XMAN=1.00000*1XMAN
IF(XMAN,LT,1.) GO TO 170
XMAN=XMAN+1.
IEXP=IEXP+1
CONTINUE
XMAN=1.*XMAN
IMSIGN=1H=
IF(IEXP,GE,0.) GO TO 180
IEXP=-IEXP
IMSIGN=1H=
IF(IEXP,GE,0.) GO TO 190
CALL NCODE(8,17H(1X41,F4.2,A1,11),IMSIGN,4,BCD)
GO TO 200
CONTINUE
CALL NCODE(8,15H(A1,F4.2,A1,I2),IMSIGN,4,BCD)
CONTINUE
CALL TSP(1,1Y1,8,BCD)

C INCREMENT THE TIC VALUE AND PROCEED
C YTIC=YTIC+DTIC
GO TO 120
C SAME PROCEDURE AS FOR Y-AXIS
C USE THE SAME LABELLING INCREMENT FOR THE X-AXIS
C
C
CONTINUE
IXL=IXL+(XTIC-IXL)*XCONVP
IF(IXL,GT,IXR) RETURN
CALL DRV(IX1,IYB,IX1,1Y2)
CALL DRV(IX1,IYS,IX1,IYT)
JRITE=IYB+9
JRITE=IX1+9
IMSIGN=1H=
IF(XTIC,GT,0.) GO TO 240
IF(XTIC,LT,0.) GO TO 230
IMSIGN=2H0.
SUBROUTINE TRACOR

C ROUTINE TO CALCULATE TURBULENCE CORRECTIONS

C ORIGINALLY WRITTEN BY HANS RUPPEL, LASL T-3
C MODIFIED BY J.L. NORTON, LASL T-3, 1975

* ----- BEGIN COMDECK PARAM ----- *
COMMON/COM/NSCP1,ITABP,ITABXP,ITABYP,IPF8,NP1,NP2,NLCP1,NLCP2,
1 NLCP3,NLCP4,IFLMSZ
* ----- END COMDECK PARAM ----- *

* ----- BEGIN COMDECK YSTORE ----- *
* ----- BEGIN COMDECK YQDIM ----- *
DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM(
15 1 ) , V ( 1 ) , V G ( 1 ) , R O ( 1 ) , S I E ( 1 ) , M P ( 1 ) , R M P ( 1 ) , R C S Q ( 1 ) , E ( 1 ) , E T I L ( 1 ) , V R O L 16 2 ( 1 ) , H ( 1 ) , R M C ( 1 ) , V P ( 1 ) , P I ( 1 ) , P L ( 1 ) , U P ( 1 ) , U T I L ( 1 ) , C O L M U ( 1 ) , E T I L ( 1 ) 17 3 1 , V L ( 1 ) , R O L ( 1 ) , A V X S V ( 1 ) , A V Y S V ( 1 ) , D L S R O I ( 1 ) , D L S R D O I ( 1 ) , C A P G A M ( 1 ) , T U G 18 4 1 , S I G ( 1 ) , U S I ( 1 ) , G R O R ( 1 ) , G R O Z C ( 1 ) , G R O P ( 1 ) , T U G V E C ( 1 ) , M T I L ( 1 ) 19 5 1 C N C ( 1 ) , C T E M P ( 1 ) , A N C U ( 1 ) , A N C V ( 1 ) , G R S V ( 1 ) , G Z S V ( 1 ) , X 1 3 K ( 1 ) , X 2 4 K ( 1 ) , 20 6 Y I 3 K ( 1 ) , X 2 4 K ( 1 ) , X R 1 3 K ( 1 ) , X R 2 4 K ( 1 ) , D K L S M ( 1 ) , A R E A ( 1 ) 21 * 22 * 23 ** BEGIN COMDECK YAGDIN **** 24 LOGICAL RESTRT , F I L M , P A P E R , T U R B 25 REAL L A H , M U 26 C COMMON / Y S C 1 / A A S C ( N 8 C P 1 ) 27 COMMON / Y S C 2 / A A S C ( 9 6 0 0 ) 28 COMMON / Y S C 2 / A A S C ( A A C ) 29 COMMON / Y S C 2 / A A S C ( A A C ) 30 COMMON / Y S C 2 / A A S C ( A A C ) 31 COMMON / Y S C 2 / A A S C ( A A C ) 32 COMMON / Y S C 2 / A A S C ( A A C ) 33 COMMON / Y S C 2 / A A S C ( A A C ) 34 COMMON / Y S C 2 / A A S C ( A A C ) 35 COMMON / Y S C 2 / A A S C ( A A C ) 36 COMMON / Y S C 2 / A A S C ( A A C ) 37 REAL M , M P , M P A R , M T I L 38 END COMDECK YAGDIN 39 ** BEGIN COMDECK YAGSC **** 40 E Q U I V A N C E ( A A S C ( 1 ) , X , X P A R ) , ( A A S C ( 2 ) , R , Y P A R ) , ( A A S C ( 3 ) , V , M P A R ) , ( 4 ) 41 A A S C ( 4 ) , U ) , ( A A S C ( 5 ) , V ) , ( A A S C ( 6 ) , R O ) , ( A A S C ( 7 ) , D F L S M , R C S Q , M P ) , ( 42 A A S C ( 8 ) , E , E T I L , A R E A , X R 1 3 K ) , ( 43 A A S C ( 9 ) , P M G O , D K L S M , R M P ) , ( A A S C ( 1 0 ) , H , R H , V P ) , ( A A S C ( 1 1 ) , P , P L , E P , U P ) , ( A A S C ( 1 2 ) , U T I L , ( 44 A A S C ( 1 3 ) , V T I L , V L , P H , P V ) , ( A A S C ( 1 4 ) , G , C Q , R O L ) , ( A A S C ( 1 5 ) , C A P S C M , U G ) , ( 45 A A S C ( 1 6 ) , T U G ) , ( A A S C ( 1 7 ) , G R O R ) , ( A A S C ( 1 8 ) , G R O Z ) , ( A A S C ( 1 9 ) , A R E A , X R 1 3 K ) , ( 46 A A S C ( 2 0 ) , R S V , G Z S V , G Z S V , G Z S V , X 1 3 K , X 2 4 K 47 A A S C ( 2 1 ) , R S V , G Z S V , G Z S V , G Z S V , X 1 3 K , X 2 4 K , X 2 4 K ) , ( 48 A A S C ( 2 2 ) , A N C U ) , ( A A S C ( 2 3 ) , A N C V ) , ( A A S C ( 2 4 ) , A N C V ) , ( A A S C ( 2 5 ) , G Z S V 49 A A S C ( 2 6 ) , G Z S V , G Z S V , G Z S V , G Z S V , A N C U , ( A A S C ( 2 7 ) , A N C V ) , ( A A S C ( 2 8 ) , A N C V ) , ( 50 A A S C ( 2 9 ) , A N C V ) , ( A A S C ( 3 0 ) , A N C V ) , ( A A S C ( 3 1 ) , A N C U ) , ( A A S C ( 3 2 ) , A N C U ) , ( 51 A A S C ( 3 3 ) , A N C U ) , ( A A S C ( 3 4 ) , A A S C ( 3 5 ) , A V X S V , X 1 3 K ) , ( A A S C ( 3 6 ) , A A S C ( 3 7 ) , A A S C ( 3 8 ) , A A S C ( 3 9 ) , A A S C ( 4 0 ) , A A S C ( 4 1 ) , A A S C ( 4 2 ) , A A S C ( 4 3 ) , A A S C ( 4 4 ) , A A S C ( 4 5 ) , A A S C ( 4 6 ) , A A S C ( 4 7 ) , A A S C ( 4 8 ) , A A S C ( 4 9 ) , A A S C ( 5 0 ) , A A S C ( 5 1 ) , A A S C ( 5 2 ) , A A S C ( 5 3 ) , A A S C ( 5 4 ) , A A S C ( 5 5 ) , A A S C ( 5 6 ) , A A S C ( 5 7 ) , A A S C ( 5 8 ) , A A S C ( 5 9 ) , A A S C ( 6 0 ) , A A S C ( 6 1 ) , A A S C ( 6 2 ) , A A S C ( 6 3 ) , A A S C ( 6 4 ) , A A S C ( 6 5 ) , A A S C ( 6 6 ) , A A S C ( 6 7 ) , A A S C ( 6 8 ) , A A S C ( 6 9 ) , A A S C ( 7 0 ) , A A S C ( 7 1 ) .
AVXS(V(IJ))=2.5*(X1+X2+X3+X4)
AVY8(V(IJ))=2.5*(Y1+Y2+Y3+Y4)
R12=IPJ+R(IPJP)
R23=IPJ+R(IPJP)
R34=IPJ+R(IPJ)
R41=IPJ+R(IPJ)
C=CONC(IJ)
SN=SIG(IJ)
RC=RO(IJ)
RCO=RO(TUQ(IJ))
RCS=RO(SIE(IJ))
AC=AREA(IJ)
IF(I.EQ.1) IMJ=IJ
IF(I.EQ.1BAR) IPJ=IJ
IF(J.EQ.2) IJM=IJ
IF(J.EQ.JP1) IJP=IJ
Z1=R12*(Y21**2+X12**2)*(SIG(IPJ)+SN)/(AREA(IPJ)+AC)
Z2=R23*(Y32**2+X23**2)*(SIG(IPJ)+SN)/(AREA(IPJ)+AC)
Z3=R34*(Y43**2+X34**2)*(SIG(IPJ)+SN)/(AREA(IPJ)+AC)
Z4=R41*(Y14**2+X14**2)*(SIG(IPJ)+SN)/(AREA(IPJ)+AC)
Z5=Z1+Z2+Z3+Z4
CTEMP(IJ)=0.5*((CONC(IPJ)-C)*Z1+(CONC(IPJ)-C)*Z2+(CONC(IPJ)-C)*Z3+
(CONC(IPJ)-C)*Z4)
OLSROI(IJ)=Z1
DLSROI(IJ)=Z2
R1=RO(IPJ)
R2=RO(IPJ)
R3=RO(IPJ)
R4=RO(IPJ)
ROL(IJ)=Z1
DLSROI(IJ)=Z1
DLSROI(IJ)=Z1
10 CONTINUE
CALL LOOP
20 CONTINUE
CALL DONE
CALL START
DO 80 J=2,JP1
DO 70 I=1,1BAR
129 IPJ=IJ+IQ
130 IMJ=IJ+IQ
131 X1=AVXSV(IPJ)
132 X2=AVXSV(IPJ)
133 X3=AVXSV(IMJ)
134 X4=AVXSV(IJM)
135 Y1=AVYSV(IPJ)
136 Y2=AVYSV(IPJ)
137 Y3=AVYSV(IMJ)
138 Y4=AVYSV(IJM)
139 P1=P(IPJ)
140 P2=P(IPJ)
141 P3=P(IMJ)
142 P4=P(IJM)
143 R01=R0(IPJ)
144 R02=R0(IPJ)
145 R03=R0(IMJ)
146 R04=R0(IJM)
147 IF(I,NE,1) GO TO 30
148 X5=X1
149 Y5=Y1
150 P3=P1
151 R03=R01
152 30 IF(J,NE,2) GO TO 40
153 X4=X2
154 Y4=Y2
155 P4=P2
156 R04=R02
157 40 IF(I,NE,IBAR) GO TO 50
158 X1=X3
159 Y1=Y3
160 P1=P3
161 R01=R03
162 50 IF(J,NE,JP1) GO TO 60
163 X2=X4
164 Y2=Y4
165 P2=P4
166 R02=R04
167 60 CONTINUE
168 RA=1.0/((X1=X3)*(Y2=Y4)*(X2=X4)*(Y1=Y3))
169 Y31=(Y3=Y1)*X1
170 Y42=(Y4=Y2)*X2
171 X13=(X1=X3)*X1
172 X24=(X2=X4)*X2
173 DPD=(P2=P4)*Y1+(P3=P1)*Y2
174 DPD=(P2=P4)*X1+(P3=P1)*X2
175 DPD=(P2=P4)*Y3+(P3=P1)*Y4
176 DPD=(P2=P4)*X3+(P3=P1)*X4
177 GRORD(IJ)=DPD+DPD+DPD+DPD
178 GRORD(IJ)=DPD
179 GRORD(IJ)=DPD
180 IJ=IPJ
181 IJM=IJM+IQ
182 IJP=IJP+IQ
183 70 CONTINUE
184 CALL LOOP
185 80 CONTINUE
CALL DONE
CALL START
CALL LOOP
DO 120 J=3,JP1
DO 110 I=1,IBAR
IMJ#IJ=NO
IMJ#IJM=NO
IF(I#EQ,1) IMJ#IJ
IF(I#EQ,1) IMJ#IJM
SIJ#0.25*(SIG(I#IJ)+SIG(IMJ)+SIG(IMJM)+SIG(IJM))
VV#0.25*(1./RVOL(I#IJ)+1./RVOL(IMJ)+1./RVOL(IMJM)+1./RVOL(IJM))#RM(
1. IJ)
X1#AVXSV(IJM)
X2#AVXSV(IJ)
X3#AVXSV(IMJ)
X4#AVXSV(IMJM)
Y1#AVYSV(IJM)
Y2#AVYSV(IJ)
Y3#AVYSV(IMJ)
Y4#AVYSV(IMJM)
IF(I#NE,1) GO TO 90
X3#X2
Y4#X1
Y5#Y2
Y4#Y1
90 CONTINUE
RXA=#1./((X1#X3)*(Y2#Y4)+(X2#X4)*(Y3#Y1))
XXA=(RO(IJM)=RO(IMJ))*(Y2#Y4)+(RO(IJM)=RO(IMJM))*(Y3#Y1)*)#RXA#SIJ
YYA=(RO(IJ)=RO(IMJM))*(X1#X3)+(RO(IMJ)=RO(IJM))*(X2#X4)*)#RXA#SIJ
YYA#YYA#VVA
IF(NCYC#EQ,NQ+1) GO TO 100
UTIL(IJ)=UTIL(IJ)+XXA#GRSV(IJ)
VTIL(IJ)=VTIL(IJ)+YYA#GZSV(IJ)
100 CONTINUE
GPSV(IJ)=XXA
GZSV(IJ)=YYA
I#IJ=NO
IJ#IJM=NO
110 CONTINUE
CALL LOOP
120 CONTINUE
CALL DONE
CALL START
DO 140 J=2,JP1
DO 130 I=1,IBAR
IPJ#IJ=NO
IPJP#JP=NO
SIJ#SIG(IJ)
DALF=.25#DT#SIJ
X13#X(IPJ)#X(IJP)
X24#X(IPJP)#X(IJ)
Y13#Y(IPJ)#Y(IJP)
Y24#Y(IPJP)#Y(IJ)
U13#U(IPJP)#U(IJ)
U24#U(IPJP)#U(IJ)
V13#V(IPJ)#V(IJP)
V24=V(IPJP)+V(IJ)
R13=(R(IPJP)+R(IJP))*DALF
R24=(R(IPJP)+R(IJP))*DALF
Z=GRRROZ(IJ)
Z=SRROR(IJ)
H1=0.5*(U13*Z+V13*ZR)
H2=0.5*(U24*Z+V24*ZR)
H3=H24*Y24*U24*ZR*Y24*R24
H4=(H13*Y13+U13*ZR*Y13)*R13

UTIL(IPJ)=UTIL(IPJP)+H3*RM(IPJP)
UTIL(IPJP)=UTIL(IPJP)+H3*RM(IPJP)
UTIL(IJP)=UTIL(IJP)+H3*RM(IPJP)
UTIL(IJP)=UTIL(IJP)+H3*RM(IPJP)

VTIL(IPJ)=VTIL(IPJP)+H1*RM(IPJP)
VTIL(IPJP)=VTIL(IPJP)+H1*RM(IPJP)
VTIL{IPJP)=VTIL{IPJP)+H2*RM(IPJP)

C *******TURBULENCE ENERGY EQUATION*******
R1J=RO(IJ)
RECHHO=RIJ/DDEL3M(IJ)
XPR1=XJ*(TUG(IJ)**2*QLEVEL)
TUQ(IJ)=RIJ*TUG(IJ)+DT*(SIJ*CAPGAM(IJ)*RIJ=GRROP(IJ)*RECHHO)+
1 DLSROQ(IJ)*RVOL(IJ)
TUQ(IJ)=(TUG(IJ)-TWTHRD*D*D*SIJ*RIJ*DT)/((1.+(XPR1+TWTHRO*O)*DT)*
1 RIJ)

C *******INTERNAL ENERGY EQUATION*******
SIE(IJ)=SIE(IJ)+DT*(DLSROI(IJ)*RVOL(IJ)*RECRHO*XPR1*TUQ(IJ))
IJM=IJM+Q
IJP=IJP+Q

SUBROUTINE TRBDIF
C ROUTINE TO CALCULATE THE ERROR FUNCTION VALUES TO BE USED
C FOR INTERPOLATION TO COMPUTE THE PARTICLE TURBULENT
C DIFFUSION MOTION
C WRITTEN BY J.NORTON, LASL T=3,1975
C BEGIN COMDECK TRBDIF
COMMON/CTDIFF/ERFV(21), NERFV, DXEF

* ~~~~ END COMDECK TBDDIF ~~~~

* ~~~~ BEGIN COMDECK YAGSC ~~~~

LOGICAL RESTART, FILM, PAPER, TURB

REAL LAM, MU

C COMMON/YSC1/AASC(NSCP1)
C COMMON/YSC2/AA(1), ANC, A0, AQPAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC,
C 1 DTO(10), DTOC(10), DTOB, DTOPOS, DT2, D2, EM10, EPS, FIPXL, FIPXR,
C 2 FIPYB, FIPY, FIXL, FIXR, FIVB, FIVT, FREZXR, GR, GROVE, GZ, CZP, I, IBAR,
C 3 IDTO, IJ, IJM, IJP, IM1, IPXL, IPXR, IPYB, IPYT, IP1, IP2, ISC2, ISC3, ITV,
C 4 IUNF, IXL, IXR, IYB, IYT, J, JBAR

COMMON/YSC2/JSCE, JP1, JP2, JP4, JUNF, JUNFO2, KXI, LAM, LPE, MU, NAME(8),
C 1 NCYC, NLC, NPS, NPT, QM, QG, QID, NQ12, NSC, NAME, ZRIG, OM, OMCY, PXCONV
C 2 , PR, FXL, PXR, PYB, PYCONV, PYT, RDT, REZRON, REZSI, REZS, RIBAR, RIBJ,
C 3 FREZYT, FREZVB, ROMFR, T, THIRD, NCLST, TOUT, TMIN

COMMON/YSC2/ITU1, ITU2, QG, TNG, TNQ, TNSV, TUSV, TURB, TTOP, PRITE, PROT,
C 1 LNLG, NILNG, TP3, TUPOT, TDGSAV, TK, T, TUDENG, EP1, SAV1, ELEVEL, TQ, IST,
C 2 VV, XCONV, XL, X, YB, YCONV, YT, YTPOLD, DTSV, DTLAST, FIBY, IYBO, YCNVL,
C 3 XCNVL, FIXRO, FIXLO, IXRO, IXLO, ISIV, JSIV, QHM, QMX, QMAX, JNM, T2, TLM,
C 4 ROMFR, ROLMYT, ROLMYB, JDUMP, TTHIRD, YE, DTR, TMSAV, DTCSAV, DTCAV, IDTV
C 5 , JDTV, IDTC, JDT, CIRC, TS, POTE, UMOM, VMDM, TMAX, TGX, JNM, T1M, JTG
C 6 , TMSAV, WMAXF, RMINEF, TSTRT
C COMMON/YSC2/ZA
C COMMON/YSC4/ITAB(ITA8P)
C COMMON/YSC4/ITAB(1000)
C COMMON/YSC5/RESTART, FILM, PAPER, IPD, IFD
C FIRST, INITIALIZE THE VARIOUS PARAMETERS

NERFV IS THE NO. OF VALUES OF ERF IN THE INTERPOLATION TABLES
NERFV=21

DXEF IS THE SPACING IN X BETWEEN THE ERF(X) TABLE ENTRIES
DXEF=WMAXF/FLOAT(NERFV=1)

COMPUTE THE TABLE VALUES

XEF=DXEF
DO 10 I=1, NERFV
XEF=XEF+DXEF
ERF(I)=ERF(XEF)
10 CONTINUE
IF(ERF(NERFV).LE.0) CALL UNCLE(4, 6HTRBERF, 26, 1 26 ERRORS IN CALCULATING ERFV)
RETURN
END

SUBROUTINE UNCLE(ISFLAG, RNAME, NCHAR, CHAR)
ERROR PROCESSING ROUTINE

ISFLAG = TYPE OF CALL
  1 = FATAL ERROR, DUMP ARRAYS (NOT IMPLEMENTED)
  2 = NORMAL EXIT
  3 = DUMP ARRAYS AND RETURN (NOT IMPLEMENTED)
  4 = FATAL ERROR, NO DUMP

RNAME = NAME OF CALLING ROUTINE
NCHAR = # OF CHARACTERS IN ERROR MESSAGE
CHAR = ERROR MESSAGE

WRITTEN BY J. L. NORTON, LASL T-3, 1974

----- BEGIN COMDECK PARAM -----
COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IFPB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMS2
----- END COMDECK PARAM -----

----- BEGIN COMDECK YAGSC -----
LOGICAL RESTRT, FILM, PAPER, TURB

REAL LAM, MU

COMMON/YSC1/AASC(NSCP1)

COMMON/YSC1/AASC(9600)

COMMON/YSC2/AA(A1), ANC, AO, AOFAC, AOM, B0, COLAMU, CYL, DR, DT, DTC, DTFC,
1 DT0(10), DTCC(10), DT02, DT06, DTPOS, DTV, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPYB, FIPYT, FIXL, FIXR, FIYB, FIYT, FRESHB, GR, GROVEL, GZ, GZP, I, IBAR,
3 IDTO, IJ, IJ3, IJ5, IM1, IMPL, IPXX, IPXY, IPYB, IPYT, IP1, IP2, ISC2, ISC3, ITV,
4 JUNF, IXL, IXR, IYB, IYT, J, JBAR

COMMON/YSC2/JCEN, JP1, JP2, JPR, JUNF02, KXI, LAM, LBP, MU, NAME(R),
1 NCYC, NLC, NPS, NPT, NQ, NQI, NQIB, NQ12, NSC, NUMITY, ZORIG, OM, OMCR, PXCONV
2 P, PXL, PXYB, PYCONV, PYT, RDT, REZRON, REZSIE, REZYB, RIBAR, RIBJB,
3 FREZYT, FRESHZ, RMFRT, T, THIRD, NCLST, TOUT, TWFIN

COMMON/YSC2/TUDI, TUSI, NCG, TNEG, TNEG5, TUSV, TURB, PTOP, PRITE, PBOH,
1 TNLG, TNLG, TP3, TUPOT, TDGSAV, TK, TI, TUPENG, EP1, SAV, QLEVEL, TO, IST,
2 VV, XCONV, XL, XR, YB, YCONV, YT, PPTOLD, DTSV, DTLAST, FIYB, IYBO, YCYNVL,
3 XCNVL, FIXR, FIXL, IXR, IXL, ISVM, JSVM, JQMN, QMX, JMAX, JMN, T2, TLM,
4 ROMFXR, ROMFYB, ROMFYB, JRUMP, TWRTHD, TE, DTR, TMASS, DTSAV, DTCSAV, IDTV
5 JOTC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC,
6 TMASSV, WMAXEF, RMINEF, TWFSTD

COMMON/YSC2/ZZ

C COMMON/YSC4/ITAB(1000)

COMMON/YSC4/ITABP

COMMON/YSC5/RESTRT, FILM, PAPER, IPD, IFD

----- END COMDECK YAGSC -----

DIMENSION CHAR(1)

DATA TP/?/,
10 WRITE(IPX, 120)
50 GO TO (60, 20, 40, 60), ISFLAG
51 20 CONTINUE
52 40 CONTINUE
56 DO 40 IPX = 6, IPD, 6
57 50 WRITE(IPX, 130)
58 GO TO 80
SUBROUTINE VELPLT(VMAX, IFLAG)

ROUTINE TO DO VELOCITY PLOTS

VMAX IS THE MAXIMUM OF THE VELOCITY CURRENTLY BEING PLOTTED IN ANY ONE DIRECTION.

IFLAG IS 0 FOR FLUID VELOCITY PLOTS SCALED BY VMAX, IT IS 1 FOR UNSCALED FLUID VELOCITY PLOTS AND 2 FOR SCALED RELATIVE VELOCITY PLOTS.

ORIGINALLY WRITTEN BY A, A, AMSDEN, LASL T-3
MODIFIED AND DOCUMENTED BY J, L, NORTON, LASL T-3 1975

--- BEGIN COMDECK PARAM ---
COMMON/PCOM/NSCP1, ITABP11TA13XP, ITABYP, IPFB, NPl, Np2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ
--- END COMDECK PARAM ---
--- BEGIN COMDECK YSTORE ---
--- BEGIN COMDECK YAQDIM ---
DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), HPAR(1), U(1), VG(1), DELSM,
2 V1(1), V2(1), R1(1), S1(1), HP(1), RMP(1), RGSQ(1), E(1), ETIL(1), RVOL,
2 P(1), PL(1), UP(1), UTL(1), UL(1), CR(1), VTL(1),
3 VL(1), RL(1), X1(1), Y1(1), X2(1), Y2(1), DLSM1(1), DLSM2(1), DLSM3(1), CAPGAM(1),
4 SIG(1), TUS(1), GROR(1), GRRO(1), GRORO(1), TUGVEC(1), MTIL(1),
5 CONC(1), CTMP(1), ANCU(1), ANCV(1), GRSV(1), GZSV(1), X13K(1), X24K(1),
6 Y13K(1), Y24K(1), XR13K(1), XR24K(1), DKL(1), AREA(1)
--- END COMDECK YAQDIM ---
--- BEGIN COMDECK YAQSC ---
LOGICAL RESTRT, FILM, PAPER, TURB

REAL LAH, MU

C COMMON/YSC1/AASC(NSCP1)
C COMMON/YSC1/AASC(9600)
C COMMON/YSC2/AASC(11), ANC, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTFAC,
C DTC, DTFAC,
C 1 DTO(10), DTOC(10), DTO2, DTPS, EVT, DTV, EM10, EPS, FIPXL, FIPXR,
C 2 FIPYB, FIPYT, FIXL, FIXR, FITY, FITY, FREQXR, GR, GRDVEL, GZ, GZP, I, IBAR,
C 3 IDTO, IJ, IJM, IPJ, IM1, IPXL, IPXR, IIPYB, IIPYT, IP1, IP2, ISC2, ISC3, ISV, ITV,
C 4 JUNF, JXL, JXR, JYB, JYT, JYD, JBAR

C COMMON/YSC2/ZCEN, JP1, JP2, JP0, JUNF, JUNFO2, KX, LAM, LPB, MU, NAME(6),
C NCYC, NLC, NPS, PTP, NO, NJ, NJB, NO2, NSC, NUMIT, ZORIG, OM, OMCYLL, PXCONV
C 1 PS, PXR, PRY, PRCONV, PRY, RDF, REDIF, REDIF, REYB, IBAR, IBAR
C 3 FREYB, FREYB, ROMFRT, T, THIRD, NCLS, TOUT, TFWIN

C COMMON/YSC3/TUSI, TUSI, NQC, TEG, TNCS, TUSV, TURB, TPT, PRTE, PBOM,
C 1 ILNG, NILNG, TP3, TUPOT, TDSAV, TK, TI, TUDENG, EP1, SAVI, QLEVEL, TO, IST,
C 2 VV, XCONV, XL, XR, YB, YCONV, YT, PTPOD, DTSV, DTLAST, FYB, FYB, YCNVL,
C 3 XCNVL, FIXO, FIXLO, IRO, ILO, ISW, JSW, QMN, QMN, WMX, WMX, WMX, WMX, T2, TLM,
C 4 ROMFRT, ROMFRT, ROMFRT, JUMP, THRD, THRD, TD, TD, TD, TD, TD, TD, TD, TD
C 5 JTY, JID, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC, JDC
C 6 JTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC
C 7 JTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC, XTC
C 8 XCNVL, FIXO, FIXLO, IRO, ILO, ISW, JSW, QMN, QMN, WMX, WMX, WMX, WMX, T2, TLM,
C 9 IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO, IDTO
C COMMON/YSC3/ZZ
C COMMON/YSC4/ITAB(ITABP)
C COMMON/YSC4/ITAB(1000)
C COMMON/YSC5/RESTR, FILM, PAPER, IPO, IFD
C * ----- BEGIN COMDECK YAEG ----- *
C * ----- BEGIN COMDECK YAEG ----- *
C EQUIVALENCE(AASC(1), X, XPAR), (AASC(2), Y, XPAR), (AASC(3), Y, XPAR),
C 1 AASC(4), U, (AASC(5), V), (AASC(6), RO), (AASC(7), DELSM), (AASC(8), MP), (AASC
C 1 (9), E, ETIL, AREA, XR3K)
C 2 (AASC(15), lXIEI), (AASC(16), PMG, OKLSTM, RMP), (AASC(9)
C 3 ), RVOL), (AASC(18), M, RH, VP), (AASC(11), P, PL, EP, UP), (AASC(12), UTIL,
C 4 1 lUL, PMX, IP), (AASC(13), lVTI, VL, PHY, PV), (AASC(14), 0, CROD), (AASC(17
C 5 ), CAPGM, UG), (AASC(18), UG), (AASC(19), SIG), (AASC(20), TUS), (AASC(21)
C 6 ), (AASC(22), GRDZ), (AASC(23), DLRDO, Y13K), (AASC(24), GZV
C 7 ), (AASC(25), DLRDO, V0), (AASC(26), GSV), (AASC(27), GRP0, TUGVEC,
C 8 Y24K), (AASC(28), MTIL), (AASC(29), CONC), (AASC(30), CTMP, XR24K),
C 9 AASC(31), ANC, (AASC(32), ANCV), (AASC(33), AVXSV, X13K), (AASC(34),
C 1 AVYSV, X24K
C REAL H, MP, XPAR, XILL
C * ----- END COMDECK YAEG ----- *
C * ----- END COMDECK YAEG ----- *
C Real COMDECK PCALL ----- *
C COMMON/PCALL/C/XCONVP, YCONVP, YUP, YLB
C * ----- END COMDECK PCALL ----- *
C * ----- END COMDECK YSTORE ----- *
C DIMENSION ITITLE(5,3)
C DIMENSION JLL2(5), JLL3(5)
C EQUIVALENCE(ITITLE(11), JLL2, JLL3,
C 1 TITLE(11), JLL3)
C DATA ITITL/4/49HFIIU, VELOCITY VECTORS SCALLED TO MAXIMUM VELOC2ITY/
C DATA ITITL/2/4/4HSCALLED FLUID VELOCITY VECTORS, 1H /
C DATA JLL3/3/MVSCALLED VELOCITY OF GRID REL. TO FLUID,
C 1 67 SCALE TO MAX. VELOC./
C C SET UP THE VELOCITY VECTOR SCALLED FACTOR,
C C VV IS 9 TIMES AN AVERAGE DR. THIS IS USED TO SCALE THE LENGTH
C C OF THE VELOCITY VECTOR SO IT IS NOT LONGER THAN THE EXTENT
C OF AN AVERAGE CELL,
DROU=VV/VMAX

SPACE FORWARD TO THE NEXT FILM FRAME

CALL ADV(I)

CALL START
DO 20 J=2,JP2
DO 10 I=1,IP1
IF(IFLAG,EQ,1) DROU=0.5*VV/SQRT(U(IJ)**2+V(IJ)**2+EM10)

(IX1, IY1) IS THE LOCATION OF THE VERTEX.

(IX2, IY2) IS THE LOCATION OF THE END OF THE VELOCITY VECTOR,
IF EITHER ARE OUTSIDE OF THE PLOTTING RECTANGLE, SKIP THIS VERTEX.

IY1=IYB+(Y(IJ)-YLB)*YCONVP
IF(IY1,GT,IYB,OR,IY1,LT,IYB) GO TO 10
IY2=IYB+(Y(IJ)+Y(IJ)*DROU-YLB)*YCONVP
IF(IY2,GT,IYB,OR,IY2,LT,IYB) GO TO 10
IX1=IXL+(X(IJ)-XL)*XCONVP
IF(IX1,GT,IXR) GO TO 10
IX2=IXL+(X(IJ)+U(IJ)*DROU)*XCONVP
IF(IX2,GT,IXR) GO TO 10

DRA W THE VECTOR

CALL DRV(IX1/IY1,IX2/IY2)

PLOT A PLUS (+) AT THE VERTEX POSITION

CALL PLT(IX1/IY1,16)

10 IJ=IJ+NO
CALL LOOP
20 CONTINUE

LABEL THE PLOT WITH VMAX

CALL LINCNT(60)
IFLGP=IFLAG+1
WRITE(IFD,40)(ITITLE(I,IFLGP),I=1,5),VMAX
WRITE(IFD,30) JNM,NAME,T,NCYC
RETURN

SUBROUTINE WIROW
ROUTINE TO WRITE ROW J FROM SCM BUFFER ONE TO LCM

ORIGINALLY WRITTEN BY A. A. AMSDEN, LASL T-3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-3, 1974

----- BEGIN COMDECK PARAM -----           COMMON/PCOM/NSCP1, ITABPX, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ

----- END COMDECK PARAM -----               COMMON/YSC1/AASC(NSCP1)
COMM/YSC2/AA(1), ANC, AR, A0FAc, A0M, BO, COAMU, CYL, DR, DT, DTC, DTFAC,
1 DTO(I0), DTOC(19), DTO2, DTOB, DTPOS, DTV, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPYB, FIPYT, FIXL, FIXR, FIYB, FIYT, FREX, GIP, GRDVEL, GZ, GZP, I, IBAR,
3 IDTO, IJ, IJM, IJP, IM, IPX, IPXY, IPY, IPYX, IPYX, IPYX, IPYX, IPYX,
4 IUNF, IXL, IXR, IYB, IYT, J, JBAR
COMMON/YSC2/JCEP, JFJ, JP2, JP4, JUNF, JUNE, XX, LAM, LPI, MUI, NAME(8),
1 NCYC, NLC, NS, NPT, NQ, NQ1, NQ2, NQ2, NQ2, NQ3, NQ3, NQ3, NQ3, NQ3,
2 NCYC, NLC, NS, NPT, NQ, NQ1, NQ2, NQ2, NQ2, NQ3, NQ3, NQ3, NQ3, NQ3,
3 NCYC, NLC, NS, NPT, NQ, NQ1, NQ2, NQ2, NQ2, NQ3, NQ3, NQ3, NQ3, NQ3,
4 NCYC, NLC, NS, NPT, NQ, NQ1, NQ2, NQ2, NQ2, NQ3, NQ3, NQ3, NQ3, NQ3,
5 JUNF, JUNE, XX, LAM, LPI, MUI, NAME(8),

COMMON/YSC2/TPUQ, TUS, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
1 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
2 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
3 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
4 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
5 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
6 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
7 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
8 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
9 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
10 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,

COMMON/YSC2/TPUQ, TUS, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
1 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
2 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
3 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
4 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
5 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
6 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
7 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
8 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
9 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
10 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,

COMMON/YSC2/TPUQ, TUS, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
1 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
2 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
3 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
4 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
5 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
6 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
7 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
8 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
9 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,
10 UI, UQI, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ, TVQ,

SUBROUTINE YAQUI2
MAIN ROUTINE FOR RUNNING CODE PHYSICS

ORIGINALLY WRITTEN BY A. A. AMSDEN AND HANS RUPPEL, LASL T-3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T-3, 1974

----- BEGIN COMDECK PARAM -----           COMMON/PCOM/NSCP1, ITABPX, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ

----- END COMDECK PARAM -----
1. \( (AT(\phi), YC0), (FT, CON) \)

* END COMDECK ASTORE *

* BEGIN COMDECK PCALL *

COMMON/PCALLC/XCONV/ YCONV/ YUP/ YLB

* END COMDECK PCALL *

COMMON/EDNST/ROTNP/ ETMP/ GONU/ CONCJ

LOGICAL EDIT, LAST, DUMP

REAL LAMO

DATA JVM, JVM, DRMIN, DZMIN, DRMAX, DZMAX, 0, 0, 0, 0 /

DATA ISWITCH/1/

C

C INITIALIZE ON STARTUP OR RESTART

C

NUMIT=0

TE=0,

DT=T,

TMASS=0,

DTVS=0,

DTC=0,

C

C INITIALIZE IF THE TURBULENCE IS ON

C

IF(.NOT., TURB) GO TO 10

TNEG=0,

TP=0,

SAV=0,

TUR=0,

TDSAV=1,

10 CONTINUE

C

C DO CERTAIN INITIALIZATION ONLY DURING RESTART

C

IF(.NOT., RESTRT) GO TO 20

CALL CINIT

IF(TURB) CALL TURBERF

GO TO 30

20 CONTINUE

C

C STARTUP, CONTINUE INITIALIZATION.

C

DTV=0,

IDTV=0,

JDT=0,

DTC=0,

10 CONTINUE

C

BEGIN LOOP OVER CYCLES

C

30 CONTINUE

C

C INITIALIZE TURBULENCE QUANTITIES

C

IF(.NOT., TURB) GO TO 40

TNEG=0, TNEG+ TNEG
125 \text{TNEG} = \theta,\\
126 \text{EP1} = \text{TK} + \text{TI} + \text{TP3} + \text{TUGENG}\\
127 \text{TIP} = \text{TI} = (\text{TMASS} - \text{TMASSV}) \times \text{REZSIE}\\
128 \text{GO CONTINUE}\\
129 \text{C} TOLD IS THE TIME AT THE END OF THE LAST CYCLE\\
130 \text{C} TOLD = \text{T2}\\
131 \text{C} IF JUST FINISHING INITIALIZATION, GO DO AN EDIT FIRST\\
132 \text{C} IF(\text{NCYC}, \text{EQ}, 0) \text{ GO TO 150}\\
133 \text{C} GO CONTINUE\\
134 \text{C} READY TO BEGIN NEXT CYCLE, INCREMENT THE CYCLE NO.\\
135 \text{C} \text{NCYC} = \text{NCYC} + 1\\
136 \text{C} SET THE NEW TIMESTEP, SAVE THE OLD ONE IN DTLAST.\\
137 \text{C} \text{DTLAST} = \text{DT}\\
138 \text{C} \text{ON THE FIRST CYCLE, SET DEFAULT DTC AND DTV}\\
139 \text{C} IF(\text{NCYC}, \text{NE}, 1) \text{ GO TO 60}\\
140 \text{C} \text{DTC} = \text{DT}\\
141 \text{C} \text{DTV} = \text{DT}\\
142 \text{C} \text{CONTINUE}\\
143 \text{C} \text{ON THE SECOND CYCLE, BOOST DT BY A FACTOR OF 10}\\
144 \text{C} IF(\text{NCYC}, \text{NE}, 2) \text{ GO TO 70}\\
145 \text{C} \text{DT} = 10 \times \text{DT}\\
146 \text{C} \text{DTSV} = \text{DT}\\
147 \text{C} \text{CONTINUE}\\
148 \text{C} \text{EXCEPT ON THE FIRST AND SECOND CYCLES, SET DT BASED ON DTV AND DTC}\\
149 \text{C} IF(\text{NCYC}, \text{GE}, 3) \text{ DT = AMIN(DTV, DTC)}\\
150 \text{C} \text{DO NOT ALLOW THE TIMESTEP TO INCREASE BY MORE THAN 25 PER CENT}\\
151 \text{C} \text{OVER THE LAST CYCLE}\\
152 \text{C} \text{DTFAC} = 1, 25\\
153 \text{C} \text{DT} = \text{AMIN(DTFAC} \times \text{DTSV}, \text{DT})\\
154 \text{C} \text{IF WE WILL BE DOING AN EDIT AFTER THE NEXT CYCLE, ADJUST DT TO}\\
155 \text{C} \text{MAKE T EXACTLY EQUAL TO THE EDIT TIME, DTPOS IS DT BEFORE}\\
156 \text{C} \text{ANY SUCH ADJUSTMENT, DTSV IS THE TIMESTEP SAVED FOR FUTURE}\\
157 \text{C} \text{REFERENCE,}\\
158 \text{C} \text{DTPOS = DT}\\
159 \text{C} \text{DTSV = DT}\\
160 \text{C} \text{IF(T + DT, GT, TOUT) DT = TOUT - T}\\
161 \text{C} \text{IF(DT, LT, 1, E8) DT = 1, E8}\\
162 \text{C} \text{UPDATE T}
TEMPORARY PATCH FOR MYLAR BALLOON CALCULATIONS

IF(T,GT,.6) ANCO=.05

ILNG IS ONE IF THIS IS AN EDIT CYCLE

ILNG=0

IF(ABS(TOUT-T),LT,EM1O) ILNG=1

SET QUANTITIES DERIVED FROM THE TIMESTEP

RDT1/T,DT

DT02=.5*DT

DT08=.125*DT

CALL YPH1

PHASE 1 - EXPLICIT LAGRANGIAN PART

CALL YPH2

PHASE 2 - PRESSURE ITERATION

CALL YPH3

PHASE 3, PART I - CONTINUOUS REZONE

CALL YFLUX


DTR1=.E30

VMAX=1.E10

CALL START

DO 110 J=2,JP2

DO 100 I=1,IP1

U(IJ)*UG(IJ)=U(IJ)

V(IJ)*VG(IJ)=V(IJ)
VMAX=AMAX1(VMAX,ABS(U(IJ)),ABS(V(IJ)))
IF(J.NE.JTM.OR.I.NE.1TM) GO TO 80
IPJ=IJ+NQ
IPJP=IPJ+NQ
XTMAX=.25*(X(IJ)+X(IPJ)+X(IJP)+X(IPJP))
YTMAX=.25*(Y(IJ)+Y(IPJ)+Y(IJP)+Y(IPJP))
80 CONTINUE
IF(J.EQ.IP1) GO TO 90
IPJ=IJ+NQ
IF(U(IJ),NE.0,.AND.V(IJ),NE.O,) DTR*AMIN1(DTR,ABS((X(IPJ)-X(IJ))/U(IJ)),ABS((Y(IJ)-Y(IJ))/V(IJ))))
90 CONTINUE
IJ=IJ+NQ
IJP=IPJ+NQ
100 CONTINUE
CALL LOOP
CALL DONE
DTR=1*DTR
SEE IF THIS IS AN EDIT CYCLE
IF(NG,NE.1) GO TO 120
YES, PLOT VELOCITY OF GRID RELATIVE TO THE FLUID,
FIY=FIYBO
IY=IYBO
YCONV=YCNVLD
XCONV=XCNVLD
FIX=FIXRO
IXL=IXL0
YLB=YB
CALL ADV(1)
CALL VELP(2)
120 CONTINUE
STORE U,V,AND RM INTO THEIR FINAL LOCATIONS AND ZERO RMP
CALL START
TPR=0,
DO 130 J=IP1,JP2
DO 130 I=1,IP1
U(IJ)=UP(IJ)
V(IJ)=VP(IJ)
RM(IJ)=RP(IJ)
TP3=TP3+GZ*Y(IJ)/RM(IJ)
RMP(IJ)=0
130 IJ=IJ+NQ
CALL LOOP
140 CONTINUE
CALL DONE
MOVE THE PARTICLES IF THERE ARE ANY
296 IF(NPT,GT,0) CALL PRTMOV
297 C FINISH THE CYCLE (PRINTS, SUMMARIES, ETC)
298 C
300 CONTINUE
301 C CALCULATE QUANTITIES TO BE USED IN AN EDIT AND FOR THE NEXT CYCLE
302 C
304 CALL CINIT
305 C T2 IS THE TIME AT THE END OF THE THIS CYCLE
306 C
308 CALL SECONO(T2)
309 C CALCULATE THE CP TIME USED PER ZONE (GRIND TIME)
310 C
312 XX=(T2-TOLD)*RIBJB
313 C CONVERT POTE TO ACTUAL GRAVITATIONAL POTENTIAL ENERGY
314 C
316 EPOT=POTE*GZ
317 C COMPUTE FIREBALL DIAMETER AND AVERAGE HEIGHT
318 C
320 PDIAM=2.*PRITE
321 PAVHT=.5*(PTOP+PBOTM)
323 C PRINT OUT CYCLE SUMMARY
324 C
325 DO 160 IPX=6,IFO,6
326 160 WRITE(IPX,340) NCYC, T, DT, T2, XX, NUMIT, CIRC, DTV, IDTV, JDTV, DTC, IDTC,
327 1 JDTC, TMAX, ITH, JTM, XTMAX, YTM, TGMX, JTG, PRITE, PTOP, PBOTM, PDIAM
328 2 ,PAVHT
329 DO 170 IPX=6,IFO,6
330 170 WRITE(IPX,330) TI, TK, EPOT, UMOM, VMOM
331 DO 180 IPX=6,IFO,6
332 180 WRITE(IPX,320) DTV, DTC
333 CALL DVMM(VMAX, JVM, DRMIN, DZMIN, DRMAX, DZMAX)
334 DO 190 IPX=6,IFO,6
335 190 WRITE(IPX,310) VMAX, JVM
337 C SEE IF IT IS TIME TO TURN ON THE TURBULENCE BASED ON TIME
338 C
339 IF(NCQ, EQ, 0, AND, T, GE, TO) GO TO 200
340 C NO, SEE IF IT IS TIME BASED ON CYCLES.
342 C
343 IF(NCYC, NE, 0, AND, NCQ, EQ, NCYC) GO TO 200
344 C NO, SKIP TURBULENCE SEEDING.
346 C
347 GO TO 240
348 200 CONTINUE
349 C
351 C YES, FLIP ON THE SWITCH.
352 C TURB=TRUE.
NCQ=NCYC

GO GET THE VORTICITY

CALL GETOMG

INITIALIZE TUG AND TUSV

TUSV=0,
CUTOFF=1.0*NQ*TUSV
CALL START
DO 220 J=2,JP1
DO 210 I=1,IBAR
TEST=TUSV*CQ(IJ)
TUSV(1J)=0.
IF(TEST.GT.CUTOFF) TUSV(1J)*TEST
TUSV=TEST
DO 210 I=1,IBAR
IJP=IJP+NQ
CALL LOOP
220 CONTINUE
CALL DONE

INITIALIZE THE PARTICLE TURBULENT DIFFUSION INTERPOLATION TABLES

CALL TRBERF

INDICATE THAT THE TURBULENCE HAS BEEN SEeded
DO 230 IPX=1,IFD
WRITE(IPX,330) NCQ,TUSV
240 CONTINUE

SEE IF TURBULENCE IS ON
IF(.NOT.,TURB) GO TO 270
YES, CALCULATE SIG AND TUS.

CALL START
DO 260 J=2,JP1
DO 250 I=1,IBAR
TUS(IJ)=TUS1
SIG(IJ)*LEVEL*TUS(IJ)*SQRT(2.*TUS(IJ))
CALL LOOP
260 CONTINUE
CALL DONE
270 CONTINUE

SEE IF IT IS TIME DO AN EDIT
EDIT=.FALSE.
IF(T+TEM10.LT.,TOUT) GO TO 280
YES, SET THE FLAG AND UPDATE OUTPUT TIME,
C EDIT=.,TRUE,
TOUT=TOUT+DTO(IDTO)
IF(T+EM10,LT,DTOC(IDTO)) GO TO 280
TOUT=DTOC(IDTO)+DTO(IDTO+1)
IDTO=IDTO+1
CONTINUE
C IF TURBULENCE HAS BEEN SEEDED, DO AN EDIT
C IF (NCYC,LE,1) EDIT=.,TRUE.
C IF THIS IS STARTUP OR THE FIRST CYCLE, DO AN EDIT
C IF (NCYC,LE,1) EDIT=.,TRUE.
C IF THE ITERATION COUNT HAS BEEN EXCEEDED, DO AN EDIT FOR
C Diagnostic Purposes
C IF (NUMIT,GE,500) EDIT=.,TRUE.
C SEE IF THIS IS THE LAST CYCLE
C LAST=.,FALSE.
C QUERY TTY
C CALL TTYTST(IFLAG)
IF(IFLAG,NE,f3) LAST=.TRUE.
IF(T,GE,TWFIN) LAST=.TRUE.
IF(T,GE,TLIM) LAST=.TRUE.
IF(NCYC,GE,NCLST) LAST=.TRUE.
IF(LAST,.AND.,IFLAG,EQ,0) EDIT=.,TRUE.
C SEE IF IT IS TIME TO DUMP
C DUMP=.,FALSE.
IF(LAST) DUMP=.,TRUE.
IF(MOD(NCYC,JDUMP),EQ,0,.AND.,NCYC,NE,0) DUMP=.,TRUE.
IF(EDIT,.AND.,NCYC,GT,1) DUMP=.,TRUE.
C DO THE EDIT IF REQUIRED
C IF(.NOT.,EDIT) GO TO 290
C IF(FILM) CALL YPLOT
C CALL YEDIT
C 290 CONTINUE
C DO THE DUMP IF REQUIRED
C IF(DUMP) CALL YDUMP
C QUIT IF THIS IS THE LAST CYCLE
C IF(LAST) RETURN
C IF THIS IS THE EDIT AFTER INITIALIZATION, IMMEDIATELY BEGIN
SUBROUTINE YARSRT

ROUTINE TO RESTART A YAGUI PROBLEM

WRITTEN BY J. L. NORTON, LABL T=3, 1975

* ----- BEGIN CONDECK PARAM ----- *
COMM/NCPOM/NSCP1, ITABP, ITABYP, ITPB, NP1, NP2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ
* ----- END CONDECK PARAM ----- *

* ----- BEGIN CONDECK YSTORE ----- *

* ----- BEGIN CONDECK YAGDIN ----- *

DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), UC(1), UG(1), DELM(1),
1 V(1), VG(1), RO(1), VSE(1), MP(1), RPM(1), RCSCQ(1), E(1), ETL(1), RYOL(1),
2 (1), AM(1), MVC(1), VPC(1), P(1), PL(1), UP(1), UT(1), UC(1), CT(1), V(1),
3 VL(1), RO(1), AVSXV(1), AVYSV(1), DLSDQ(1), DLSRQ(1), DPAGA(1), VUQ(1),
4 (1), SIG(1), TUS(1), GRROR(1), GRROZ(1), GRROP(1), TUPVEC(1), MTIL(1),
5 CONC(1), CTEM(1), ANCP(1), ANCV(1), GRBV(1), GZBV(1), X13K(1), X24K(1),
6 Y13K(1), Y24K(1), XR13K(1), XR24K(1), DLKSM(1), AREA(1)

* ----- END CONDECK YAGDIN ----- *

* ----- BEGIN CONDECK YAGSC ----- *

LOGICAL RESTR, FILM, PAPER, TURB

REAL LAM, MU

COMMON/YSC1/AASC(NSCP1)
COMMON/YSC1/AASC(9600)

COMMON/YSC2/AAC(1), AN, AO, ADFAC, ASH, B0, COLAMU, CYLC, DR, DT, DTC, DTFAC,
1 DTO(I0), DTOR(I0), DTO2, DTO5, DT0, DTO, DZ, EM10, EPS, FIPXL, FIPXR,
2 FIPvB, FIPVY, FIPX, FIPY, FIPZ, FIPF, FREZXR, GR, GRDVEL, GZ, GZP, I, ISAR,
COMMON/YSC2/JEN, JP1, JP2, JPO, JUNF, JUNFO2, KX, LAH, LPB, MU, NAME(R),
1 NCYC, NL, NPS, NPT, NO, NQ1, NQ12, NSC, NUMIT, ZORIG, OM, OMCL, PXCONV
2 PXL, PXR, PYB, PXCONV, PYT, RDT, REZRD, REZSIE, REZYB, RIBAR, RIBJB,
3 FREZYT, FREZBY, ROMFY, T, THIRD, NCLST, TOUT, TKFIN
4 COMMON/YSC2/TOUI, TUSI, NCO, TNEQ, TNEGTY, TUSV, TUSB, TTOP, PRTE, PBTOM,
5 I, ILG, NILG, TP, TUPOT, TDQSAV, TK, T, TUDENG, EP, SAV1, GLEVEL, TO, IST,
6 TUSV, XCONV, XL, XR, YB, YCONV, YT, TPTPOL, DTSV, TLAST, FLYBO, FLYBO, YCNWLO,
7 XCNWLO, FPRXO, FPRXO, FPRXO, FLXO, ISW, JSW, GNM, GMM, WM, XNM, T2, TLIM,
8 RMPFXR, ROMFYT, JDUMP, JDT, ORRS, TDR, TMASS, DTVSAV, DTSAV, IDTV
9 5, JDT, IDTC, JDT, JUTC, JAR, JAC, JAC, JAC, JAC, JAC, JAC, JAC, JAC, JAC, JAC, JAC,
10, THMSSV, WMXXEF, RMINEF, TSTRTD

COMMON/YSC2/ZZ

COMMON/YSC4/ITAB(IITYP)

COMMON/YSC4/ITAB(1000)

COMMON/YSC5/RESTRY, FILM, PAPER, IPO, IFD

* ----- END COMDECK YASC -----  
* ----- BEGIN COMDECK YASE -----  

REAL M, MP, MPAR, TIL

* ----- END COMDECK YAPE ----- 

* ----- END COMDECK YSTORE ----- 

INTEGER TAPE, AA

DIMENSION RCYCLE(3,3)

DATA (RCYCLE(1), 1, 1), 1/9*0/

DATA ISTATUS/0/

TLIMSV=TLIM

SET UP NAMELIST INPUT TABLE

ASSIGN 120 TO IERRT

CALL TABDEF(RCYCLE, 6HRCYCLE, 3, IERRT)

CALL TABSET(RCYCLE, 6HRCYCLE, ITCYCLE, IEFLAG, 0, 0, 0)

CALL TABSET(RCYCLE, 4HTAPE, TAP, IEFLAG, 0, 0, 0)

READ THE RESTART TAPE NO. AND THE RESTART CYCLE. ITCYCLE=1

SIGNIFIES RESTART FROM THE LAST DUMP ON THE TAPE.

ITCYCLE=1

IF(IETYL, NE, 0) CALL UNCLE(4, 6HYRST, 27)
OPEN THE INPUT DUMP FILE
CALL OPEN(7,1)
NO ERRORS, CALL ROUTINE TO GET TAPE IF TAPE NO. WAS READ.
(ONLY FUNCTIONAL ON CROS/7600)
IF(TAPE,NE,0) CALL GETTPE(TAPE)
RECEIVE 7
JNSC=LOCF(ZZ)-LOCF(AA)+1
READ THE NEXT CYCLE ON THE TAPE
CONTINUE
CALL SCBUFF(AA,JNSC,7,0,1,IERROR)
CHECK FOR ERRORS
IF(IERROR,GE,0) GO TO 30
CALL UNCLE(1,6HYARSRT,18,18HSCBUFF INPUT ERROR)
CONTINUE
IF(IERROR,NE,0) GO TO 40
SEE IF TERMINAL DUMP RECORD WAS FOUND
IF(AA(1),EQ,666) GO TO 70
CONTINUE
NO, PRINT CYCLE NO. OF DUMP LAST READ,
DO 50 IPX=6,1FD,6
WRITE(IPX,130) NCYC
SEE IF LAST CYCLE IS DESIRED
IF(INTCYC,EQ,(-1)) GO TO 60
NO, SEE IF WE HAVE FOUND THE CORRECT CYCLE.
IF(NCYC=INTCYC) 60,80,110
CONTINUE
CORRECT CYCLE NOT FOUND YET, SKIP REST OF DUMP AND GO READ NEXT,
CALL RTAPE
GO TO 10
DUMP TAPE TERMINATION FOUND, FATAL UNLESS INTCYC IS -1.
IF(INTCYC,NE,(-1)) CALL UNCLE(4,6HYARSRT,17,17HEOF ON INPUT TAPE)
OK, DUMP WAS THE LAST ONE READ.
SUBROUTINE VASET
C
ROUTINE TO GENERATE A NEW PROBLEM
C
C ORIGINALLY WRITTEN BY A.A.AMSDELLASL T=3
C MODIFIED AND DOCUMENTED BY J.L.NORTON,LASL T=3,1974
C
* ----- BEGIN COMDECK PARAM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK PARAM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YSTORE ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YSTORE ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- END COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
10 1 NLCP3,NLCP4,IFLMSZ
11 * ----- BEGIN COMDECK YADDIM ----- COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
15 1) V(1), VG(1), PD(1), SIL(1), MP(1), RMP(1), RCSQ(1), E(1), ETIL(1), RVOL
16 2) (1), RM(1), VP(1), P(1), PL(1), UP(1), UTIL(1), UL(1), CO(1), VTI(1)
17 3) VL(1), ROL(1), AVXSV(1), AVSV(1), DLSR01(1), DLSRQG(1), CAPGM(1), TILQ
18 4) (1), SIG(1), TUS(1), GRR0(1), GRROZ(1), GRROQ(1), QUVEC(1), MTIIL(1),
19 5) CONC(1), CTEMP(1), ANCU(1), ANCV(1), GRSV(1), GZSV(1), X13X(1), X24K(1),
20 6) Y13K(1), Y24K(1), XR13K(1), XR24K(1), DKL0H(1), AREA(1)
21 * ===== BEGIN COMDECK YAOQ3 =====
22 * logical RESTR(1), FILM, PAPER, TURB
23 REAL LAM, MU
24 C COMMON/YSC1/AASC(NSCP)
25 COMMON/YSC1/AASC(9600)
26 COMMON/YSC2/AA(1), ANC, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTFAC,
27 1 DTO(10), DTOC(10), DT02, DT08, DTPOS, DTV, DZ, EMO, EPS, EPF, FIPX, FIPXR,
28 2 FIPYB, FIPYT, FIXL, FIXR, FYIB, FITY, FRELX, GR, GROVEL, GZ, GZP, ISR, IBAR
29 3 IDTO, IJ, IJM, IJP, IM, IPXL, IPXR, IPYB, IPYT, IP1, IP2, ISC2, ISC3, IV,
30 4 IUNF, IXL, IXR, IYR, IBAR
31 C COMMON/YSC2/FCEN, JP1, JP2, JPU, JUNF, JUNPO2, KXI, LAM, LPB, MU, NAME(B)
32 1 NCYC, NLC, NPT, NQ1, NQ2, NSC, NUMIT, ZORIG, OH, OMCL, PXCONV
33 2 PXL, PXR, PYB, PYCONV, PXY, ROT, REZR(1), RESIE, REZYB, RIBAR, RIBJB
34 3 FRELX, FRELX, FROX, TR, AIL, ACL,拐, NCLST, TOUT, THFIN
35 C COMMON/YSC2/TUQI, TUSI, NCQ, TNEQ, TNEG, TNEQ(1), TUSV, TURB, TPIQ, PRTE, PRB, PRB,
36 1 PTV, PTV, TVP, TVP, TVT, TTV, TVT, TTV, TTV, TVT, TTV, TVT, TTV, TTV, TTV, TTV, TTV,
37 2 TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV,
38 3 TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV, TVV,
39 C COMMON/YSC2/TUQJ, TUQJ, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
40 1 NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ, NCQ,
41 2 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
42 3 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
43 4 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
44 5 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
45 6 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
46 7 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
47 8 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
48 9 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
49 0 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
50 1 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
51 2 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
52 3 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
53 4 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
54 5 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
55 6 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
56 7 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
57 8 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
58 9 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
60 1 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
61 2 NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF, NCF,
62 * ===== BEGIN COMDECK YAOQ3 =====
63 * logical RESTR(1), FILM, PAPER, TURB
64 REAL LAM, MU
65 C COMMON/YSC4/ITAB(1000)
66 COMMON/YSC4/ITAB(1000)
67 COMMON/YSC5/RESTR, FILM, PAPER, IPD, IFD
68 * ===== BEGIN COMDECK YOOG =====
69 * logical RESTR(1), FILM, PAPER, TURB
70 REAL LAM, MU
71 C READ THE PROBLEM TITLE
READ(5,320) NAME
C
INITIALIZE THE PROBLEM VARIABLES
CALL DEFINE
READ THE INPUT VARIABLES
CALL YINPUT
OUTPUT THE PROBLEM TITLE
DO 10 IPX=6,IFD,6
10 WRITE(IPX,320) NAME
IF NCQ,LT,0 THERE WILL BE NO TURBULENCE
IF(NCQ,LT,0) GO TO 70
THERE WILL BE TURBULENCE, PRINT OUT THE INPUT QUANTITIES
DO 20 IPX=6,IFD,6
20 WRITE(IPX,290) QLEVEL,TUQI,TU9I,NCQ,TQ,TSTAR,TSTTD,WMAXEF,RMINEF
GO TO 50
THERE WILL BE NO TURBULENCE, INDICATE SUCH,
30 CONTINUE
DO 40 IPX=6,IFD,6
40 WRITE(IPX,300)
CONTINUE
PRINT GENERAL INPUT VARIABLES
DO 60 KT=6,IFD,6
WRITE(KT,330) IRAR,JBAR,IUNF,JUNF,JCN,JDI,CYL,GRDVEL,A0,A0M,B0,
KXI
WRITE(KT,340) MU,LAM,OM,EP8,GR,GZ
WRITE(KT,350) FREZXR,FREZYT,FREZYB,ZORIG,YB,REZYO,REZTON,REZSIE
WRITE(KT,360) GZP
WRITE(KT,370) TDT,NCLST,TMFIN,PAPER,FILM
WRITE(KT,380) ANC,A0FAC
WRITE(KT,390) DT0(N),N=1,10
WRITE(KT,390) DT0C(N),N=1,10
CONTINUE
C
CALCULATE AND STORE PROBLEM CONSTANTS
I=1(IBAR=1
IP1=IBAR+1
IP2=IBAR+2
JP1=JBAR+1
JP2=JBAR+2
JP4=JBAR+4
RIBAR=1./FLOAT(IBAR)
RIBJB=1./FLOAT(IBAR*JBAR)
NOI is the number of words of data needed to store one row, it is the size of a small core buffer.

NOI = NOIP1

See routine START for definitions.

ISC2 = NOI + 1
ISC3 = ISC2 + NOI

See routine STARTD for definitions.

ITV = JP1 + NOI

NSC is the number of words of small core in common YSC2

NSC = LOC(F(ZZ)) - LOC(F(AA)) + 1

NLC is the number of words of large core used in common YLC1, it should not exceed the parameter value.

NLC = JP4 + NOI

Go check the large and small core parameters.

CALL PCHK

IDTO is the subscript in the DT0C array such that
DT0C(IDTO) = (LT, T, LE, DT0C(IDTO))

IDTO = 1

TOUT is the time at which output should occur next

TOUT = DTOC(IDTO)

IF(TOUT, GT, T) GO TO 90

70 CONTINUE

90 CONTINUE

The timestep for the first cycle is reduced by a factor of 10 from the specified initial DT.
DTPOS is the timestep that is possible based on stability criteria, the actual timestep DT may be less if it has been adjusted on an output cycle so that TOUT exactly.

DTPOS=DT

DTSV is the timestep from the last cycle

DTSV=DT

NCYC is the cycle no.

NCYC=0

COLAMU is 1/(2/3*(2*MU+LAM))

COLAMU=1.*5/(LAM+MU+MU+EM10)

NILNG is the no. of times a particle has been stored for use in making time-dependent particle plots

NILNG=0

IUNF must be at least 1

IUNF=MAX(IUNF,1)

JUNF must be at least 2

JUNF=MAX(JUNF,2)

JUNFD2=JUNF/2

IF JCEN is zero, set it to JBAR/2

IF(JCEN.EQ.0) JCEN=JBAR/2

GO generate the particles

CALL PRTGEN

GO generate the mesh

CALL MSHMKR

GO set the plot quantities

CALL FILMCO

**************************************************

LOCATE which cell each particle is in if there are any particles

**************

IF(NPT.LE.0) GO TO 210

ASSUMING that the mesh is still rectangular, store the x and y values in XTAB and YTAB, make sure the arrays are large enough.
IF(JP2,GT,ITABYP) CALL UNCLE(4,SHYASET,21,21HXTAB ARRAY OVERFLOWED
1)
IF(IP1,GT,ITABXP) CALL UNCLE(4,SHYASET,21,21HXTAB ARRAY OVERFLOWED
1)
CALL START
DO 120 J=2,JP2
XTAB(J) = Y(IJ)
IF(J,GT,2) GO TO 110
DO 100 I=1,IP1
XTAB(I) = X(IJ)
110 CALL LOOP
120 CONTINUE
C INITIALIZE THE LCM ADDRESS AND THE PARTICLE COUNT
C IECP=1
C NPPT=0
C BRING IN A BUFFER=FULL OF PARTICLE DATA
C LPB IS SET IN PARTGEN
C 130 CALL ECRD(AASC, NLCP1+IECP-1,LPB,IDUM)
C KP=1
C CONTINUE
C LOCATE THE I AND J OF THE CELL CONTAINING THE PARTICLE IN QUESTION
C DO 150 J=2,JP2
C IF(VTAB(J),GT,YPAR(KP)) GO TO 170
C 150 CONTINUE
C DO 160 IPX=6,IFD,6
C WRITE(IPX,270) YTAB(JP2),YPAR(KP),KP
C CALL UNCLE(1,SHYASET,23,23HJ OF PARTICLE NOT FOUND)
C 170 CONTINUE
C DO 180 IPX=6,IFD,6
C IF(XTAB(I),GT,XPAR(KP)) GO TO 200
C 180 CONTINUE
C CALL UNCLE(1,SHYASET,23,23HJ OF PARTICLE NOT FOUND)
C 200 CONTINUE
C INCREMENT THE PARTICLE COUNT
C NPPT=NPPT+1
C I AND J OF CELL FOUND, CODE IT AND STORE IN ITAB, FIRST MAKE SURE
C THAT STORAGE WILL NOT BE OVERFLOWED,
C IF(NPPT,GT,ITABP) CALL UNCLE(4,SHYASET,21,21HITAB ARRAY OVERFLOWED
C 1)
C ITAB(NPPT)*(J-2)+IP1+I=1
C SEE IF ALL PARTICLES HAVE BEEN PROCESSED
300 IF(NPPT,GE,NPT) GO TO 210
301 C NO, SEE IF THE BUFFER NEEDS TO BE RELOADED.
302 C
303 C KP=KP+3
304 C IF(KP,LT,LPB) GO TO 140
305 C
306 C YES, INCREMENT THE LCM POINT AND RELOAD THE BUFFER
307 C
308 C IECP=IECP+LPB
309 C GO TO 150
310 C
311 210 CONTINUE
312 C
313 C **********************************************
314 C CALCULATE THE CELL-CENTERED VOLUMES (RECIPIROCALS), MASSES, AND
315 C TOTAL ENERGIES FOR ALL CELLS
316 C **********************************************
317 C
318 C CALL START
319 DO 230 J=2,JP1
320 DO 220 I=1,IBAR
321 IPJ=I+NP
322 IPJP=IPJ+NP
323 X1=XS(IPJ)
324 Y1=YS(IPJ)
325 R1=RS(IPJ)
326 X2=X1(IPJP)
327 Y2=Y1(IPJP)
328 R2=RS(IPJP)
329 X3=X2(IPJP)
330 Y3=Y2(IPJP)
331 R3=RS(IPJP)
332 X4=X3(IPJP)
333 Y4=Y3(IPJP)
334 R4=RS(IPJP)
335 RVOL(IPJP)=RVOL(IPJ)=0
336 *(IPJP))*1.25*(U(IPJP)**2+U(IPJP)**2+U(IPJP)**2+U(IPJP)**2+V(IPJP)**2)
337 1 **2*V(IPJP)**2*V(IPJP)**2*V(IPJP)**2)
338 I=IPJP
339 220 IJP=IPJP
340 CALL LOOP
341 230 CONTINUE
342 CALL DONE
343 C
344 C **********************************************
345 C COMPUTE THE VERTEX MASSES, EXCEPT FOR BOUNDARY VERTICES, THE VERTEX
346 C MASS IS JUST 1/4 OF THE MASSES OF ALL THE CELLS HAVING THE
347 C VERTEX AS A CORNER,
CALL STARTD
DO 250 JJ=2, JP2
IF (JP2 .EQ. JJ) GO TO 251
IMJ = IJM - NQ
IMJ = IJM - ND
XX = 0.0
IF (I .NE. IP1 .AND. J .NE. 2) XX = M(IJM) + XX
IF (I .NE. IP1 .AND. J .NE. JP2) XX = XX + M(IJM)
IF (I .NE. IP1 .AND. J .NE. 2) XX = XX + M(IJM)
RM(IJ) = XX
IJM = INJ14
CALL LOOPD
250 CONTINUE
RETURN

SUBROUTINE YDUMP
C ROUTINE TO DO A YAQUI DUMP
C WRITTEN BY J.L. NORTON, LASL T=3, 1975
END
* ****** BEGIN COMDECK PARAM ******
   COMMON/PCOM/NSCP1,ITABP,ITABXP,ITABYP,IPFB,NP1,NP2,NLCP1,NLCP2,
   1 NLCP3,NLCP4,ITLMS
* ****** END COMDECK PARAM ******
1 ****** BEGIN COMDECK YADIM ******
DIMENSION X(1),XPAR(1),R(1),YPAR(1),Y(1),MPAR(1),U(1),UG(1),DELSH(1)
   1,V(1),VG(1),RO(1),SZ(1),MP(1),RMP(1),RCSG(1),E(1),ETIL(1),RVL
   2,MP(1),MP(1),MP(1),MP(1),P(1),PL(1),UP(1),UTILITY(1),UL(1),C(1),VTIL(1)
   3,VL(1),ROL(1),AVSXV(1),AVSV(1),DLSRO(1),DLSRO1(1),CAPGAM(1),TUQ
   4,1,SIG(1),TUS(1),GROR(1),GRORO(1),GRORO1(1),TUGVEC(1),MTIL(1),
   5,CONC(1),CTEMP(1),ANCE(1),ANCE(1),GRSV(1),GRSV(1),X13K(1),X24K(1),
   6,Y13K(1),Y24K(1),Y13K(1),X24K(1),DKLSM(1),AREA(1)
* ****** END COMDECK YADIM ******
21 ****** BEGIN COMDECK YAGSC ******
   LOGICAL RESTRT,FILM,PAPER, TURB
   REAL LAM,MU
   COMMON/YSC1/AASC(NSCP1)
   COMMON/YSC1/AASC(9600)
   COMMON/YSC2/AASC(41),ANC,AR,ABFAC,ABH,BO,COLAMU,CYL,DR,DT,DTF,FAC,
   1,DTC(10),DTOS(10),DTOS,DTOS,DTOS,DTOS,DTOS,DTOS,DTOS,DTOS,DTOS,
   2,DIPY,DIPY,DIPY,DIPY,DIPY,DIPY,DIPY,DIPY,DIPY,DIPY,
   3,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   4,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   5,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   6,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   7,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   8,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   9,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   10,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   11,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   12,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   13,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   14,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   15,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   16,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   17,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   18,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   19,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   20,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   21,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   22,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   23,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   24,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   25,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   26,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
   27,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,3-DIPY,
DATA TP/0,/

C PRINT TIMING FIGURE

CALL SECOND(TP)
WRITE(59,70) TP,T,NCYC

C PRINT THE TIME AND CYCLE BEING DUMPED

DO 10 IPX=6,IFO,6
  10 WRITE(IPX,90) T,NCYC

C BACKSPACE OVER THE TRAILER RECORD WRITTEN BY THE LAST DUMP
C BACKSPACE 8
C WRITE OUT THE SCM COMMON
JNSC=LOCF(ZZ)=LOCF(AA)+1
CALL SCBUFF(AA,JNSC,8,1,1,IERROR)
IF(IERROR,EQ,0) GO TO 30
  20 CALL UNCLE(4,5HYDUMP,19,19HSCBUFF OUTPUT ERROR)
C I/O SUCCESSFULLY COMPLETED
C 30 CONTINUE
C WRITE OUT THE ARRAYS FROM LCM
CALL LCBUFF(0,NLC,8,1,1,IERROR)
C CHECK FOR ERRORS IN LCBUFF
IF(IERROR,EQ,0) GO TO 50
C LCBUFF ERROR, KILL THE RUN,
C 40 CALL UNCLE(4,5HYDUMP,19,19HLCBUFF OUTPUT ERROR)
C NO ERRORS, CONTINUE
C 50 CONTINUE
C SEE IF THERE ARE ANY PARTICLES
IF(NPT.LE.0) GO TO 60
C YES, WRITE OUT THE PARTICLE ARRAYS FROM LCM,
CALL LCBUFF(NLCP1,NPS,8,1,1,IERROR)
IF(IERROR,NE,0) GO TO 60
C WRITE OUT THE ITAB ARRAY FROM SCM
CALL SCBUFF(ITAB,ITABP,8,1,1,IERROR)
IF(IERROR,NE,0) GO TO 20
SEE IF THERE IS ANY TIME-DEPENDENT PARTICLE DATA

IF(NILNG.LE.0) GO TO 60

YES, WRITE IT OUT FROM LCM,

CALL LCBuff(NLCP1+NLCP2,2*NP1*NILNG,0,1,IERROR)

IF(IERROR,NE,0) GO TO 40

TERMINATE THE DUMP WITH A SPECIAL TRAILER RECORD

CALL SCBuff(IJunk,1,8,1,IERROR)

IF(IERROR,NE,0) GO TO 20

PRINT TIMING FIGURE

WRITE(N9,80) TP

RETURN

70 FORMAT(1H ,20HBEGIN YDUMP AT CP = ,F10.4,1H, 5X15HPROBLEM TIME = ,
1 PE12.5,1H, 9X8HCYCLE = ,IS)
80 FORMAT(1H ,18HEND YDUMP AT CP = ,F10.4)
90 FORMAT(1H ,15HTAPE DUMP AT T=,1PE12.5,8H, CYCLE=,IS)

END

----------------------------------------------

SUBROUTINE YEDIT

ROUTINE TO PRINT FULL LISTING OF YAGUI MESH QUANTITIES

ORIGINALLY WRITTEN BY A,A,AMSDEH, LASL T=3
MODIFIED AND DOCUMENTED BY J.L.NORTON, LASL T=3, 1974

* ----- BEGIN COMDECK PARAM ----- *
COMMON/Pcom/NLCP1,NLCP2,ITABP,ITABYP,IPFB,ITABP,IPFB,NLCP1,NLCP2,
IFLSZ
* ----- END COMDECK PARAM ----- *

* ----- BEGIN COMDECK YSTORE ----- *

* ----- BEGIN COMDECK YAODIM ----- *
DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPA(1), U(1), UG(1), DELSM(1),
V(1), W(1), RO(1), SIE(1), MP(1), RMP(1), RCOL(1), E(1), ETIL(1), RVOL(1),
PL(1), MP(1), P(1), PL(1), U(1), UTIL(1), UL(1), CO(1), Y(1), U(1),
VL(1), VL(1), AVXSV(1), AVCYSV(1), DLSROI(1), DLSROQ(1), CAPGAM(1), TUG(1),
SIG(1), TUS(1), GROR(1), GROD(1), TUGVEC(1), HTIL(1),
MRU(1), ETMP(1), ANCU(1), ANCV(1), GRSVC(1), GRSVP(1), X15K(1), X24K(1),
6 Y15K(1), Y24K(1), X13K(1), X20K(1), DKLSM(1), AREA(1)
* ----- END COMDECK YAODIM ----- *

* ----- BEGIN COMDECK YAOSC ----- *
DATA TP/9/
DATA INDEF/1777000000000077777778/
CALL SECOND(TP)
WRITE(59,1200) TP
CALL C
C
CALL 60 LINES PER PAGE OF DATA. SET THE COUNT TO FORCE A TOP-OF-PAGE HEADING FIRST THING
DO 90 I=1,IP1
C SEE IF A HEADER NEEDS TO BE PRINTED
91 C IF(LINESF,LT,60) GO TO 30
92 C YES, RESET THE LINE COUNT
93 C LINESF=0
94 C PRINT THE HEADER
95 C DO 20 IPX=1,IPDO,6
96 C WRITE(IPX,140) JNM,NAMET,NCYC
97 C IF(TURB) GO TO 10
98 C WRITE(IPX,150)
99 C GO TO 20
100 CONTINUE
101 CONTINUE
102 CONTINUE
103 IPJM=IJM+NQ
104 IPJ=IJ+Q
105 DO=0
106 PRM=0
107 IF(TURB) GO TO 60
108 TEST=RM(IJM)
109 IF(ITEM,NE,INDEF) GO TO 40
110 IF(ITEM,NE,0,) PRM=1./TEST
111 CONTINUE
112 TEST=RVOL(IJM)
113 IF(ITEM,NE,INDEF) GO TO 50
114 IF(ITEM,NE,0,) PRV=1./TEST
115 CONTINUE
116 D=CONC(IJM)
117 C TEMPORARY PATCH FOR OUTPUTTING VORTICITY
118 C D=CONC(IJM)
119 PRM=CM(IJM)
120 GO TO 70
121 PRV=TUS(IJM)
122 D=CONC(IJM)
123 PRM=TUS(IJM)
124 70 CONTINUE
125 DO 80 IPX=1,IPD,6
126 WRITE(IPX,170) I,J,X(IJM),Y(IJM),U(IJM),V(IJM),6E(IJM),RD(IJM),
127 1 PRV,0,PRM,P(IJM)
128 LINESF=LINESF+1
129 IJ=IPJ
130 IJM=IPJM
131 CALL LOOP
132 100 CONTINUE
133 IF(.NOT.FILM) GO TO 110
134 FYB=FYBG
135 IYB=IYBD
136 YCONV=YCNVL
SUBROUTINE YEXIT(IABORT)

YAQUI ERROR RECOVERY ROUTINE

WRITTEN BY J. L. NORTON, LASL T-3, 1975

CALL EXIT

END

SUBROUTINE YFLUX

ROUTINE TO FLUX MASS, MOMENTUM, AND ENERGY IF THE GRID VELOCITY IS NOT EQUAL TO THE FLUID VELOCITY

ORIGINALLY WRITTEN BY A. A. AMSDEN AND HANS RUPPEL, LASL T-3

MODIFIED BY J. L. NORTON, LASL T-3, 1975

* ----- BEGIN COMDECK YSTORE ----- *

5 CONC(1),CTEMP(1),ANCU(1),ANCV(1),GRSV(1),GSV(1),X13K(1),X24K(1),
6 V13K(1),V24K(1),XR13K(1),XR24K(1),DKLSM(1),AREA(1)
* ----- END COMDECK YAGDIM ----- *
* ----- BEGIN COMDECK YAGSC ----- *
LOGICAL RESTRT,FILM,PAPER,TURB
REAL LAM,MU
COMMON/YSC1/AASC(NSCP1)
COMMON/YSC1/AASC(NSCP1)
COMMON/YSC2/AASC(1),ANG,ABFAC,AML,BM,COLAMU,CYL,DR,DT,DTB,DTFAC,
1 DTO(10),DTOL(10),DTOR,DTPOS,DTV,DTZ,EM19,EP,FTPL,FTPX,FTPY,
2 FTPYB,FIPYT,FIXL,FIXR,FIXY,FIYT,FREZXR,GR,GROVEL,GZ,GZP,I,IBAR,
3 IDTO, IJMJ, IJPP, IMI, IPXL, IPXR, IPYW, IPYT, IPX, IP2, ISC, ISC3, ITV,
4 IUN, IX, IY, IYB, IYT, J, JBAR
COMMON/YSC2/JCN,JP1,JP2,JP4,JP7,JP9,KXI,KLAM,LPM,MU,NAME(8),
1 NCYC, NC, NPS, NPT, NQ, NOI, NOI, NQ, NUCY, NUC, NQ, NUM, NUMS, NUMT,
2 PCL, PXR, PXB, PXCV, PYT, RGT, REZCN, REZDIE, REZFD, RIBBAR, ROBAR,
3 FREZ, FREZY, ROMPF, T, TMD, NCLST, TOUT, TWF
COMMON/YSC2/TUO, TUO, NCO, TNEG, TNEGV, TUOV, TURB, PTP, PRT, P50, TST,
1 ILN, ILN, TPS, TPS, TDSAV, TK, TI, TIVG, TP1, SAV1, QLEVEL, TQ, IST,
2 VV, VXCON, XL, XR, YB, YCONV, Y, YF1, YBO, YBO, YCNVL,
3 XCNVL, FXLC, IXRO, IXL, ISW, JSW, QMN, QMX, XMAX, XMIN, T2, TLM
4 ROMX, ROMFYT, ROMFVB, JUMP, TMHD, TE, DFR, TA, DTS, DTVSAV, DTS, IDT
5 JOTV, IOTC, JOTC, JRC, JTI, POTE, JUM, VMOM, TMAX, TMX, ITM, JTG
6 THASSV, WHMAEF, RMINEF, T83RTD
COHt4DN1YSC2/ZZ
COHt4DN1YSC2/ZZ
COMMON/YSC2/ITABP(1000)
COMMON/YSC2/ITABP(1000)
* ----- END COMDECK YAGSC ----- *
* ----- BEGIN COMDECK YAGEO ----- *
EQUIVALENCE(AASC(1),ANG,ABFAC,AML,BM,COLAMU,CYL,DR,DT,DTB,DTFAC,
1 DTO(10),DTOL(10),DTOR,DTPOS,DTV,DTZ,EM19,EP,FTPL,FTPX,FTPY,
2 FTPYB,FIPYT,FIXL,FIXR,FIXY,FIYT,FREZXR,GR,GROVEL,GZ,GZP,I,IBAR,
3 IDTO, IJMJ, IJPP, IMI, IPXL, IPXR, IPYW, IPYT, IPX, IP2, ISC, ISC3, ITV,
4 IUN, IX, IY, IYB, IYT, J, JBAR
COMMON/YSC2/JCN,JP1,JP2,JP4,JP7,JP9,KXI,KLAM,LPM,MU,NAME(8),
1 NCYC, NC, NPS, NPT, NQ, NOI, NOI, NQ, NUCY, NUC, NQ, NUM, NUMS, NUMT,
2 PCL, PXR, PXB, PXCV, PYT, RGT, REZCN, REZDIE, REZFD, RIBBAR, ROBAR,
3 FREZ, FREZY, ROMPF, T, TMD, NCLST, TOUT, TWF
COMMON/YSC2/TUO, TUO, NCO, TNEG, TNEGV, TUOV, TURB, PTP, PRT, P50, TST,
1 ILN, ILN, TPS, TPS, TDSAV, TK, TI, TIVG, TP1, SAV1, QLEVEL, TQ, IST,
2 VV, VXCON, XL, XR, YB, YCONV, Y, YF1, YBO, YBO, YCNVL,
3 XCNVL, FXLC, IXRO, IXL, ISW, JSW, QMN, QMX, XMAX, XMIN, T2, TLM
4 ROMX, ROMFYT, ROMFVB, JUMP, TMHD, TE, DFR, TA, DTS, DTVSAV, DTS, IDT
5 JOTV, IOTC, JOTC, JRC, JTI, POTE, JUM, VMOM, TMAX, TMX, ITM, JTG
6 THASSV, WHMAEF, RMINEF, T83RTD
COHt4DN1YSC2/ZZ
COHt4DN1YSC2/ZZ
COMMON/YSC2/ITABP(1000)
COMMON/YSC2/ITABP(1000)
* ----- END COMDECK YAGEO ----- *
* ----- BEGIN COMDECK YAGEO ----- *
REAL M, MP, MPAR, MQTT
* ----- END COMDECK YAGEO ----- *
* ----- BEGIN COMDECK YAGEO ----- *
COMMON/YSC/PCOMMON/NSCP1, ITABP, ITABXP, ITABYP, IPFB, IPN1, IPN2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ
* ----- END COMDECK PARAM ----- *
COMMON/YSC/PCOMMON/NSCP1, ITABP, ITABXP, ITABYP, IPFB, IPN1, IPN2, NLCP1, NLCP2,
1 NLCP3, NLCP4, IFLMSZ
* ----- BEGIN COMDECK PARAM ----- *
COMMON/AASC/ATC(100), FT(100)
DIMENSION IX(1,1), IY(1,1), IZ(1,1), YCO(1), YCO(1), CON(1)
EQUIVALENCE(At, IX1, IY1, IZ1, YC01, YCO1, T, CON(1)
1 AT(9), YCO(FT, CON)
* ----- END COMDECK PARAM ----- *
DTC==16,E30
CALL START
DO 50 J=2,IPJ
DO 40 I=1,IBAR
I1=I+1
IPJ=IPJ+NQ
IPJP=IPJP+NQ
X1=X(IPJ)
Y1=Y(IPJ)
R1=R(IPJ)
X2=X(IPJP)
Y2=Y(IPJP)
R2=R(IPJP)
X3=X(IPJP)
Y3=Y(IPJP)
R3=R(IPJP)
UL1=UL(IPJ)
VL1=VL(IPJ)
UL2=UL(IPJP)
VL2=VL(IPJP)
UL3=UL(IPJP)
VL3=VL(IPJP)
UL4=UL(IPJP)
VL4=VL(IPJP)
UD1=UG(IPJ)=0.5*(UL1+U(IPJ))
VD1=VG(IPJ)=0.5*(VL1+V(IPJ))
UD2=UG(IPJP)=0.5*(UL2+U(IPJP))
VD2=VG(IPJP)=0.5*(VL2+V(IPJP))
UD3=UG(IPJP)=0.5*(UL3+U(IPJP))
VD3=VG(IPJP)=0.5*(VL3+V(IPJP))
UD4=UG(IPJP)=0.5*(UL4+U(IPJP))
VD4=VG(IPJP)=0.5*(VL4+V(IPJP))
X12=X1+X2
X23=X2+X3
X34=X3+X4
X41=X4+X1
Y12=Y1+Y2
Y23=Y2+Y3
Y34=Y3+Y4
Y41=Y4+Y1
R12=R1+R2
R23=R2+R3
R34=R3+R4
R41=R4+R1
UL1=UL1+UL2
UL2=UL2+UL3
UL3=UL3+UL4
UL4=UL1+UL4
VL1=VL1+VL2
VL2=VL2+VL3
VL3=VL3+VL4
VL4=VL1+VL4
D=.25*RVOL(IJ)*(R12*(U12+Y12+V12+X12)+R23*(U23+Y32+V23+X23)+R34*(U34+Y43+V34+X34)+R41*(U41+Y41+V41+X41))
VOLR=VOLT+VOLC=1./RVOL(IJ)
IF(I,NE,IBAR) VOLR=1./RVOL(IPJ)
IF(J,NE,JP1) VOLT=1./RVOL(IPJ)
IF(I,EQ,1) GO TO 60
FL=FR
AL=AAR
10 IF(J,EQ,2) GO TO 70
FB=FT(I)
AB=AT(I)
20 FR=DT08*A12*(UD1*UD2)*Y21*(VD1*VD2)*X12
AR=A0M*SIGN(1.,FR)+BO*2.*FR/(VOLR+VOLC)
FT(I)=DT08*R23*(UD2*UD3)*Y32*(VD2*VD3)*X23
AT(I)=AOM*SIGN(1.,FT(I))+BO*2.*FT(I)/(VOLT+VOLC)
40 XX=MAX1(ABS(FB),ABS(FT(I)),ABS(FL))
DTC=MIN1(DTC,DTPOS*AR/XY*RVOL(IPJ)+DTPOS*ABS(D)*EM10)
IF(DTCSAV,NE,DTC) IOTC=I
IF(DTCSAV,NE,DTC) JOTC=J
DTCSAV=DTC
MP(IJ)=MTIL(IJ)*VOLC+FR*(1.+AR)*ROL(IPJ)+FT(I)*(1.+AT(I))
2 AL+ROL(IMJ)+FB*(1.+AB)*ROL(IPJ)+(1.+AB)*ROL(IMJ)
50 RDE=RO(IJ)*ETIL(IJ)
81E(IJ)=1./MP(IJ)*RDE+VOLC+FR*(1.+AR)*RDE+(1.+AR)*RO(IPJ)*ETIL(IJ)
1 IPJ)+FT(I)*(1.+AT(I))=RO+1.+AT(I))*RO(IPJ)*ETIL(IPJ)+FL*(1.+AB)*RO(IJ)
2 +AL)*ROE+1.+AL)*ROI(IPJ)*ETIL(IPJ)+FB*(1.+AB)*ROE+(1.+AB)*RO(IJ)
3 IJ)*ETIL(IJ)),
IF(.NOT.TURB) GO TO 30
56 RDE=RO(IJ)*TUGQ(IJ)
157 TUGQVEC(IJ)=1./MP(IJ)*RQE*VOLC+FR*(1.+AR)*RQE+(1.+AR)*RO(IPJ)*TUGQ(IJ)
1 IPJ)+FT(I)*(1.+AT(I))=RQE+1.+AT(I))*RO(IPJ)*TUGQ(IPJ)+FL*(1.+AB)*ROE(IJ)
2 +AL)*RQE+1.+AL)*ROI(IPJ)*TUGQ(IPJ)+FB*(1.+AB)*RQE+(1.+AB)*ROI(IPJ)
3 TUGQ(IJ))
30 CONTINUE
162 RDE=RO(IJ)*CONC(IJ)
163 CTEMP(IJ)=VOLC+RQE+FR*(1.+AR)*RQE+(1.+AR)*CONC(IPJ)+FT(I)*(1.+AT(I))
1 +ROG+1.+AT(I))=CONC(IPJ)+FL*(1.+AL)*ROE+(1.+AL)*CONC(IMJ)
165 2 )+FB*(1.+AB)*RO+1.+AB)*CONC(IPJ)
166 CTEMP(IJ)=CTEMP(IJ)*RVOL(IJ)
167 RVOL(IJ)=0./X(R1*R2*R3*R4)*(X1*X3)*(Y2*Y4)*(Y1*Y3)*(X2*X4))
168 IJP
169 IJP=IPJP
40 IJMP=IJP+1
171 CALL LOOP
90 CONTINUE
173 CALL DNE
174 GO TO 80
175 DO 100 I=IBAR
176 AL=A0M*SIGN(1.,FL)+BO*2.*FL*RVOL(IJ)
GO TO 77
178 FB=DT08*R43*(UD3*UD4)*Y34*(VD3*VD4)*X34
179 AB=A0M*SIGN(1.,FB)+BO*2.*FB*RVOL(IJ)
GO TO 80
80 CALL START
181 DO 110 J=2,JP1
183 DO 100 I=1,IBAR
184 RO(IPJ)=MP(IJ)*RVOL(IJ)
185 CONC(IPJ)=CTEMP(IJ)
186 IF(J,EQ,2) RO(IPJ)=ROL(IPJ)
IF (.NOT. TURB) GO TO 90

TUG(IJ) = TUGVEC(IJ)

IF (TUG(IJ)**LT.0.) TNEG = TNEG + TUG(IJ) * RO(IJ) / RVOL(IJ)

IF (TUG(IJ)**LT.0.) TUG(IJ) = 0.

190 CONTINUE

192 IF (J .EQ. JP1) RO(IJP) = ROL(IJP)

193 IF (I .EQ. IBAR) RO(I+NG) = ROL(I+NG)

194 IJM = IJM + NG

195 IJP = IJP + NG

196 100 IJ = IJ + NG

197 CALL LOOP

198 110 CONTINUE

199 CALL END

200 CALL START

201 DO 150 J = 2, JP2

202 JJP = JJ

203 DO 120 II = 1, IP1

204 IIP = II

205 IMJ = J + NG

206 IMJM = IMJ + NG

207 XX = 0.

208 IF (I .NE. IP1, AND, J .NE. JP2) XX = XX + P(IJ)

209 IF (I .NE. IP1, AND, J .NE. JP2) XX = XX + P(IJ)

210 IF (I .NE. J, AND, J .NE. JP2) XX = XX + P(IJM)

211 IF (I .NE. J, AND, J .NE. JP2) XX = XX + P(IMJ)

212 RMP(IJ) = XX

213 IJM = IMJM

214 120 IMJM = IMJM

215 CALL LOOPD

216 130 CONTINUE

217 CALL START

218 DO 150 J = 2, JP2

219 DO 140 II = 1, IP1

220 XX = XX + P(IJ) / RM(IJ)

221 UP(IJ) = XX * UL(IJ)

222 VP(IJ) = XX * VL(IJ)

223 IJ = IJ + NG

224 CALL LOOP

225 150 CALL DONE

226 DO 250 J = 1, IBAR

227 IPJP = IPJ + NG

228 X = X(IPJ)

229 Y = Y(IPJ)

230 R = R(IPJ)

231 UL = UL(IPJ)

232 VG = VG(IPJ)

233 X2 = X(IPJP)

234 Y2 = Y(IPJP)

235 R2 = R(IPJP)

236 UL2 = UL(IPJP)

237 VG2 = VG(IPJP)

238 250 CONTINUE
240  Vl2=Vl(IPJP)
241  Vg2=Vg(IPJP)
242  X3=x(IJP)
243  Y3=y(IJP)
244  R3=r(IJP)
245  UL3=UL(IPJP)
246  UG3=UG(IPJP)
247  VL3=VL(IPJP)
248  VG3=VG(IPJP)
249  X4=x(IJ)
250  Y4=y(IJ)
251  R4=r(IJ)
252  UL4=UL(IJ)
253  UG4=UG(IJ)
254  VL4=VL(IJ)
255  VG4=VG(IJ)
256  XX=x(IJP)
257  UL13=0.5*(UL1+UL3+UL(IPJP)+UL(IPJP))
258  UL24=0.5*(UL2+UL4+UL(IPJP)+UL(IPJP))
259  F13=x(xR3)*((UG1+UG3=UL13)*(Y3=Y1)+(VG1+VG3=VL13)*(X1=X3))
260  F24=x(xR2)*((UG2+UG4=UL24)*(Y2=Y4)+(VG2+VG4=VL24)*(X4=X2))
261  FM2=F13*RM(IJP)
262  FM3=F24*RM(IJP)
263  XC=25*(X1+X2+X3+X4)
264  YC=25*(Y1+Y2+Y3+Y4)
265  UC=25*(UL13+UL24)
266  VGC=25*(UG1+UG2+UG3+UG4)
267  UGCUC=UGC=UC
268  VGVC=VGC=VC
269  A=UGCUC*(Y3=Y1)+VGVC*(X1=X3)
270  B=UGCUC*(Y4=Y2)+VGVC*(X2=X4)
271  IF(A) 160,180,170
272  GO TO 240
273  IF(B) 210,200,200
274  IF(B) 210,190,200
275  GO TO 240
276  IF(A) 160,180,170
277  GO TO 240
278  IF(B) 210,200,200
279  IF(B) 210,190,200
280  GO TO 240
281  160  Ws=H=0.5
282  170  Ws=(YC=Y4)*UGCUC+(X4=X3)*VGVC/((Y3=Y4)*UGCUC+(X4=X3)*VGVC)
283  180  Ws=(YC=Y4)*UGCUC+(X4=X3)*VGVC/((Y3=Y4)*UGCUC+(X4=X3)*VGVC)
284  190  Ws=H=0.5
285  200  GO TO 240
286  210  Ws=1,
287  Hs=(Y3=YC)*UGCUC+(X3=XC)*VGVC/((Y3=Y2)*UGCUC+(X3=X3)*VGVC)
288  220  Ws=1,
289  Hs=(YC=Y1)*UGCUC+(X1=XC)*VGVC/((Y2=Y1)*UGCUC+(X1=X2)*VGVC)
290  230  Hs=0,
291  Hs=(Y3=Y4)*UGCUC+(X4=X3)*VGVC/((Y3=Y4)*UGCUC+(X4=X3)*VGVC)
292  240  DMW=1,w
293  250  GO TO 240
294  260  DMH=1,h
295  270  UB=x=UL2+DMW=UL3+DMH=UL1+DMH=UL4
SUBROUTINE YINIT

ROUTINE TO INITIALIZE THE CODE

WRITTEN BY J.L. NORTON, LASL T-5, 1974

* ----- BEGIN COMDECK PARAM ----- *
COMMON/P/COM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2, 1 NLCP3, NLCP4, IFILMSZ
* ----- END COMDECK PARAM ----- *

----------- BEGIN COMDECK YSTORE -----------

* ----- END COMDECK YAGDIM ----- *

* ----- BEGIN COMDECK YAGSC ----- *
LOGICAL REBRT, FILM, PAPER, TURB
REAL LAM, HU
COMMON/YSC1/AASC(9600)
COMMON/YSC2/AA(1), ANC, AN, AOFAC, A0H, B0, COLAH, CYL, DR, DT, DTC, DTFAC, 1 DTN(10), DTOC(10), DTO2, DTO3, DTV, DT, DTM, EM10, EPS, FIPX, FIPYR, 26
READ THE JOB PARAMETERS. FIRST SET UP FOR NAMELIST INPUT.

ASSIGN 30 TO IERRT
CALL TABDEF(START,5HSTART,5,IERRT)
CALL TABBSET(START,6HRESTR,RESTR,IEFLAG,0,0,0)
CALL TABBSET(START,4HFILM,FILM,IEFLAG,0,0,0)
CALL TABBSET(START,5HPAPER,PAPER,IEFLAG,0,0,0)
CALL TABBSET(START,6HWRAPUP,WRAPUP,IEFLAG,0,0,0)

DO THE ACTUAL READ

RESTRT = IF .TRUE., THE CODE WILL BE RESTARTED FROM A DUMP TAPE (DEFAULT)
         = IF .FALSE., THE PROBLEM WILL BE GENERATED FROM INPUT
PAPER = IF .TRUE., OUTPUT WILL OCCUR ON PAPER (DEFAULT=FALSE)
        = IF .TRUE., OUTPUT WILL OCCUR ON FILM (DEFAULT=TRUE)
WWRAPUP = NO. OF CP SECONDS TO ALLOW FOR PROBLEM TERMINATION
              AFTER THE LAST CYCLE (DEFAULT=20)

CALL NAMELIST(START,5,IEFLAG)

IF(IEFLAG,NE;0) CALL UNCLE(U,5HINIT,23,23HERROR IN START NAMELIST 1)

NO ERRORS, SET THE OUTPUT DO LOOP INDICES,

SET UP THE CONSTANT 1/3
THIRD=1/3.
SET UP THE CONSTANT 2/3
TWTHIRD=2*THIRD
GET THE JOB ID
CALL GETJOB(JNM)
GET THE JOB TIME LIMIT
CALL GETJTL(TLIM)
GIVE A LITTLE TIME FOR WRAPUP
TLIM=TLIM-WRAPUP
SUBROUTINE YINPUT

ROUTINE TO READ YAQUI INPUT VARIABLES

WRITTEN BY J.L. NORTON, LASL T-301975

* * * BEGIN COMECK YSTORE * * *

* * * BEGIN COMECK YADIM * * *

DIMENSION XI(1), XPAR(1), R(1), YPAR (1), M(1), U(1), UG(1), NELS(1),
1 1, 2, 3, 4, 5, X(1), V(1), W(1), H(1), S(1), Q(1), P(1), PL(1), UP(1),
1 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
1 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
1 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
1 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38,
1 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
1 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74,
1 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92,
1 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
1 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122,
1 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136,
1 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150,
1 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164,
1 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178,
1 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192,
1 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206,
1 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220,
1 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234,
1 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248,
1 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262,
1 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276,
1 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290,
1 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304,
32 1 ILNG, NILNG, TP3, TUPOT, TDOSAV, TK, TI, TUGENG, EP1, NAV1, QLEVEL, TQ, IST, 33 2 VV, XCONV, XL, XR, YB, YCONV, YTT, TPTPOD, DTSV, DTLAST, FLYBO, YBO, YCNVL, 34 3 XCNVL, FIXRO, FIXLD, IXRO, IXL, I5V, JSBV, QMN, QMX, HMAX, JNH, T2, TLIM, 35 4 ROMFXR, ROMFYR, ROMFYB, JDUMP, XTMDRO, TE, DTR, TMASS, DTVSAV, DCXSAG, IDTV 36 5, JDTV, IDTC, JDTC, CIRC, TIS, POTE, UMOM, VMOM, TMAG, TGMX, ITH, JTM, ITG, JTG 37 6, TMASSV, MMAXEF, RMINEF, TSTTD 38 COMMON/YSCC/2Z 39 C COMMON/YSC4/ITABC (ITABP) 40 COMMON/YSC4/ITAB (1000) 41 COMMON/YSC5/RESTRT, FILM, PAPER, IPO, IFD 42 * ===== END COMDECK YASC =========== 43 * ===== BEGIN COMDECK YASEQ ========= 44 EQUIVALENCE (AASC(1), X, XPAR), (AASC(2), Y, XPAR), (AASC(3), Y, MPAR), 45 1 AASC(4), U, (AASC(5), V), (AASC(6), RO), (AASC(7), DELSM, RC8Q, MP), (AASC 46 1 (8), ETIL, AREA, XR(13)K), 47 2 (AASC(15), SIE), (AASC(16), PHB, DKL8M, RMP), (AASC(9) 48 3 ), (VOL), (AASC(10), M, RM, VP), (AASC(11), P, PL, EP, UP), (AASC(12), UTL, 49 4 UL, PMX, PRU), (AASC(13), VTL, VL, PMY, PV), (AASC(14), Q, CG, ROL), (AASC(17) 50 5 ), (CAPGMV, LUG), (AASC(18), TU0), (AASC(19), SIG), (AASC(20), TUS), (AASC( 51 6 21), GRORR), (AASC(22), GRROD), (AASC(23), DLSROI, Y13K), (AASC(24), GS8V 52 7 ), (AASC(25), DLSROI, VG), (AASC(26), GR8V), (AASC(27), GRORP, TEOVEC, 53 8 Y24K), (AASC(28), MTIL), (AASC(29), CONC), (AASC(30), CTEMP, XR24K), ( 54 9 AASC(31), ANC), (AASC(32), ANCV), (AASC(33), AVXSV, X13K), (AASC(34), 55 1 AVSV, X24K) 56 REAL M, MP, MPAR, MTIL 57 * ===== END COMDECK YAGED =========== 58 * ===== END COMDECK YSTORE ========= 59 DIMENSION CARDN(320,9) 60 DATA (CARDN(II), II=1,147)/147*0/ 61 DATA IEFLAG/0/ 62 C SET UP THE NAMELIST INPUT TABLE 63 C ASSIGN 00 TO IERRT 64 CALL TABDEF (CARDN, 5HSHARDN, 0, IERROR) 65 CALL TABSET (CARDN, 5HUN, MU, IEFLAG, 0, 0, 0, 0) 66 CALL TABSET (CARDN, 5HLAM, LAM, IEFLAG, 0, 0, 0, 0) 67 CALL TABSET (CARDN, 5HOM, OM, IEFLAG, 0, 0, 0, 0) 68 CALL TABSET (CARDN, 5HEPS, EPS, IEFLAG, 0, 0, 0, 0) 69 CALL TABSET (CARDN, 5HGR, GR, IEFLAG, 0, 0, 0, 0) 70 CALL TABSET (CARDN, 5HGEZ, GEZ, IEFLAG, 0, 0, 0, 0) 71 CALL TABSET (CARDN, 5HREZXR, REZXR, IEFLAG, 0, 0, 0, 0) 72 CALL TABSET (CARDN, 5HREZYR, REZYR, IEFLAG, 0, 0, 0, 0) 73 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 74 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 75 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 76 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 77 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 78 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 79 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 80 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 81 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 82 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 83 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 84 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 85 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 86 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 87 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0) 88 CALL TABSET (CARDN, 5HREZRON, REZRON, IEFLAG, 0, 0, 0, 0)
CALL TABSET(CARDN,4HIBAR,IBAR,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HJBAR,JBAR,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HIUNF,UNF,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HJUNF,JUNF,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HCEN,JCEN,IEFLAG,0,0,0)
CALL TABSET(CARDN,2HDZ,DZ,IEFLAG,0,0,0)
CALL TABSET(CARDN,3HCYL,CYL,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HGRDVEL,GRDVEL,IEFLAG,0,0,0)
CALL TABSET(CARDN,2HA0,A0,IEFLAG,0,0,0)
CALL TABSET(CARDN,3HA0M,A0M,IEFLAG,0,0,0)
CALL TABSET(CARDN,2HB0,B0,IEFLAG,0,0,0)
CALL TABSET(CARDN,3HKXI,KXI,IEFLAG,0,0,0)
CALL TABSET(CARDN,3HANC,ANC,IEFLAG,0,0,0)
CALL TABSET(CARDN,5HA0FAC,FAC,IEFLAG,0,0,0)
CALL TABSET(CARDN,5HDT0,DTO,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HDT0C,DT0C,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HFRZYT,FRZYT,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HFRZYL,FRZYL,IEFLAG,0,0,0)
CALL TABSET(CARDN,5HZORIG,ORIG,IEFLAG,0,0,0)
CALL TABSET(CARDN,5HZJUMP,JUMP,IEFLAG,0,0,0)
CALL TABSET(CARDN,4HNAME,NAM,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HMNAME,NAM,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HRMINF,MINF,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HTSTRD,TSTRD,IEFLAG,0,0,0)
CALL TABSET(CARDN,6HIEOF,IEOF,IEFLAG,0,0,0)

C READ THE INPUT VARIABLES

10 CONTINUE

IEOF=0
CALL NAMEST(CARDN,5,IEFLAG)

C CHECK FOR INPUT ERRORS

IF(IEFLAG,NE,0) CALL UNCLE(4,6HYINPUT,26,
1 26HCARDN NAMELIST INPUT ERROR)

C NO ERRORS, CHECK FOR EOF.

IF(IEOF,NE,0) GO TO 20

C NO, CONTINUE READING.

GO TO 10

20 CONTINUE

C CHECK THE VALUES OF THE INPUT PARAMETERS

IF(MU,LT,0.0) CALL UNCLE(4,6HYINPUT,8,6MMU,LT,0.0)
IF(LAM,LT,0.0) CALL UNCLE(4,6HYINPUT,9,9HLAM,LT,0.0)
IF(OM,LT,0.0) OR(OM,GT,2.0) CALL UNCLE(4,6HYINPUT,10,10ILLEGAL OM)
IF(EPS,LE,0.0) CALL UNCLE(4,6HYINPUT,9,9HEPS,LE,0.0)
IF(FREZXR,LT,1.0) CALL UNCLE(4,6HYINPUT,12,12HFREZXR,LT,1.0)
IF(FREZYL,LT,1.0) CALL UNCLE(4,6HYINPUT,12,12HFREZYL,LT,1.0)
IF(FREZYB,LT,1.0) CALL UNCLE(4,6HYINPUT,12,12HFREZYB,LT,1.0)
IF(REZRON,LE,0.0) CALL UNCLE(4,6HYINPUT,12,12REZRON,LE,0.0)
SUBROUTINE YPH1

ROUTINE TO DO THE EXPLICIT LAGRANGIAN HYDRO (PHASE 1)

ORIGINALLY WRITTEN BY A. A. AMSDEN, LASL T=3
MODIFIED AND DOCUMENTED BY J. L. NORTON, LASL T=3, 1975

* ------- BEGIN COMDECK YSTORE -------
* ------- BEGIN COMDECK YADDIM -------
* DIMENSION X(1), XPAR(1), R(1), YPAR(1), Y(1), MPAR(1), U(1), UG(1), DELSM(1)
* 1, V(1), VG(1), RO(1), OE(1), HP(1), RMP(1), RCSQ(1), E(1), ETIL(1), RVOL
* 2, (1), M(1), ANC(1), VPC1(1), P1(1), PL(1), UP(1), UTIL(1), UL(1), CG(1), YTIL(1)
* 3, VL(1), ROL(1), AVXS(1), AVYS(1), DLSSQM(1), DLSSQR(1), CAPGAM(1), TUG
* 4, SIG(1), TUS(1), GRROR(1), GRROZ(1), GROP(1), TUGVEC(1), MTIL(1),
* 5 CONC(1), CTMP(1), ANC(1), ANCV(1), GR8V(1), QZSV(1), X3K(1), X24K(1),
* 6 YSK(1), Y2K(1), XR13K(1), XR24K(1), DKL8M(1), AREA(1)
* ------- END COMDECK YADDIM -------
* ------- BEGIN COMDECK YAD48C -------
* LOGICAL RESTRY, FILM, PAPER, TURB
* REAL LAM, MI

COMMON/YSC11AASC(N8CP1)
COMMON/YSC1/AASC(9600)

COMMON/YSC2/AASC(1),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC3/AASC(3),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC4/AASC(5),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC5/AASC(7),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC6/AASC(9),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC7/AASC(11),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC8/AASC(13),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC9/AASC(15),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC10/AASC(17),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC11/AASC(19),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC12/AASC(21),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC13/AASC(23),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC14/AASC(25),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC15/AASC(27),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC16/AASC(29),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC17/AASC(31),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC18/AASC(33),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC19/AASC(35),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC20/AASC(37),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC21/AASC(39),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC22/AASC(41),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC23/AASC(43),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC24/AASC(45),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC25/AASC(47),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC26/AASC(49),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC27/AASC(51),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC28/AASC(53),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,

COMMON/YSC29/AASC(55),ANC,ANZ,ABFAC,ABM,ACLAM,AYL,DR,DT,DTFAC,
IPJ = IJ + NO

XX IS ZERO IF I IS 1 OR IP1, 1 OTHERWISE
YY IS ZERO IF J IS 2 OR JP2, 1 OTHERWISE

XX = 1, 
YY = 1.

U1, V1 ARE VELOCITIES AT VERTEX (I=1, J) UNLESS I=1 IN WHICH CASE VERTEX (I, J) IS USED

IF(I, EQ, 1) GO TO 10
U1 = U(IJ)
V1 = V(IJ)
GO TO 20

10 XX = 0, 0
U1 = U(IJ)
V1 = V(IJ)

U2, V2 ARE VELOCITIES AT VERTEX (I+1, J) UNLESS I=IP1 IN WHICH CASE VERTEX (I, J) IS USED

20 IF(I, EQ, IP1) GO TO 30
U2 = U(IPJ)
V2 = V(IPJ)
GO TO 40

30 U2 = U(IJ)
V2 = V(IJ)
XX = 0, 0

U3, V3 ARE VELOCITIES AT VERTEX (I, J+1) UNLESS J=2 IN WHICH CASE VERTEX (I, J) IS USED

40 IF(J, EQ, 2) GO TO 50
U3 = U(IJM)
V3 = V(IJM)
GO TO 60

50 U3 = U(IJ)
V3 = V(IJ)
YY = 0, 0

U4, V4 ARE VELOCITIES AT VERTEX (I, J+1) UNLESS J=JP2 IN WHICH CASE VERTEX (I, J) IS USED

60 IF(J, EQ, JP2) GO TO 70
U4 = U(IJP)
V4 = V(IJP)
GO TO 80

70 YY = 0, 0
U4 = U(IJ)
V4 = V(IJ)

U5, V5 ARE VELOCITIES AT VERTEX (I, J)

80 U5 = U(IJ)
V5 = V(IJ)
THE NODE COUPLER IS APPLIED TO THE U (V) VELOCITY IF FLAGU (V) IS 1.

FLAGU=0,
FLAGV=0.

SET FLAGU AND FLAGV, THE TECHNIQUE IS TO EXAMINE THE THREE VERTICES (I=1,J), (I,J), AND (I+1), OR (I,J-1), (I,J),AND (I,J+1). IF THE VELOCITY OF VERTEX (I,J) IS THE MINIMUM OR MAXIMUM OF EITHER TRIPLET, THE NODE COUPLER WILL BE APPLIED, OTHERWISE, IT WILL NOT BE APPLIED.

IF(U5, EQ, AMAX1(U1, U2, U5), OR, U5, EQ, AMIN1(U1, U2, U5)) FLAGU=1.0
IF(V5, EQ, AMAX1(V1, V2, V5), OR, V5, EQ, AMIN1(V1, V2, V5)) FLAGV=1.0
IF(U5, EQ, AMAX1(U3, U4, U5), OR, U5, EQ, AMIN1(U3, U4, U5)) FLAGU=1.0
IF(V5, EQ, AMAX1(V3, V4, V5), OR, V5, EQ, AMIN1(V3, V4, V5)) FLAGV=1.0

UAV, VAV ARE THE DIFFERENCES OF THE AVERAGE VELOCITIES OF THE NEIGHBORING VERTICES AND THE VELOCITIES OF VERTEX (I,J) ITSELF.

UAV=0.25*(U1+U2+U3+U4)-U5
VAV=0.25*(V1+V2+V3+V4)-V5

CALCULATE THE BODY FORCE ACCELERATIONS, THE SECOND TERM IS THE NODE COUPLER IF ONE IS TO BE APPLIED.

ANCX=1.0*FLAGU*UAV
ANCY=1.0*FLAGV*VAV
AX=GR+ANCX
AY=GZ+ANCY

UPDATE THE VERTEX VELOCITIES DUE TO THE BODY FORCES AND THE NODE COUPLER. NOTE THAT THE WALLS ARE RIGID.

UTIL(I,J)=U(I,J)+DT*AX)*XX
VTIL(I,J)=V(I,J)+DT*AY)*YY

SAVE THE PART OF THE VELOCITY CHANGE DUE TO THE NODE COUPLER

ANCU(I,J)=ANCX*XX*DT
ANCY(I,J)=ANCY*YY*DT

I=IPJ
IPJ=IPJ+NQ
90 IJM=IJM+NQ
CALL LOOP
100 CONTINUE
102 CALL DONE

CALCULATE THE CHANGE TO THE VERTEX VELOCITIES DUE TO THE PRESSURE TERM IN THE MOMENTUM EQUATION, LOOP OVER ALL CELLS, NOT VERTICES.

CALL START
DO 160 J=2,JP1
DO 160 I=1,IBAR
ROL(I,J)=R.
IPJ=IPJ+NQ
107 RETURN
108 END
IPJP=IJP+NO
VERTEX (I+1,J) IS LABELED 1, (I+1,J+1) IS 2, (I,J+1) IS 3, AND (I,J) IS 4

X1*X(IPJ)
Y1*Y(IPJ)
R1*R(IPJ)
U1*U(IPJ)
V1*V(IPJ)
X2*X(IPJP)
Y2*Y(IPJP)
R2*R(IPJP)
U2*U(IPJP)
V2*V(IPJP)
X3*X(IPJP)
Y3*Y(IPJP)
R3*R(IPJP)
U3*U(IPJP)
V3*V(IPJP)
X4*X(IJ)
Y4*Y(IJ)
R4*R(IJ)
U4*U(IJ)
V4*V(IJ)

X(Y)NM = X(Y)n*x(y)h
X2*X2=X4
Y2*Y2=Y4
X3*X3=X1
Y3*Y3=Y1

UOR IS THE AVERAGE U VELOCITY DIVIDED BY THE AVERAGE X POSITION
OF THE VERTICES, FOR CYLINDRICAL GEOMETRY, FOR SLAB GEOMETRY
UOR IS ZERO,
UOR=(U1+U2+U3+U4)/(R1+R2+R3+R4)*CYL
HRMN IS .5*(RN+RN)
HR13=.5*(R1+R3)
HR24=.5*(R2+R4)

D(T2M)=D(T2)/(MASS OF VERTEX N)
D(T2M1)=D(T2*R(IPJ)
D(T2M2)=D(T2*R(IPJP)
D(T2M3)=D(T2*R(IJP)
D(T2M4)=D(T2*R(IJ)

XY IS THE CROSS PRODUCT OF THE DIAGONALS OF THE CELL
XY=X24*Y31-X31*Y24
CAREA IS THE CELL AREA WHICH IS JUST HALF OF THE CROSS PRODUCT
OF THE DIAGONALS
CAREA = 0.5 * XY

SAVE THE AREA FOR USE IN THE TURBULENCE CALCULATION IN TRBCOR

AREA(IJ) = CAREA

RXY IS THE RECIPROCAL OF XY (HALF THE RECIPROCAL OF THE CELL AREA)

RXY = 1 / XY

U(V) IJ - U(V) I = U(V) J

U24 = U2 = U4
U31 = U3 = U1
V24 = V2 = V4
V31 = V3 = V1

DU(V) OR(Z) IS THE DIFFERENCE FORM FOR THE DERIVATIVE OF THE RADIAL (AXIAL) VELOCITY WITH RESPECT TO R (Z) AT THE CELL CENTER

DUDR = RXY * (U24 * Y31 - U31 * Y24)
DUDY = RXY * (X24 * U31 - X31 * U24)
DVDR = RXY * (V24 * Y31 - V31 * Y24)
DVDY = RXY * (X24 * V31 - X31 * V24)

COMPUTE DIV(VELOCITY) = VOLUME DILATATION

D = (DUDR + DVDR) * (1, * UDR + DT) + UDR

SAVE THE VELOCITY DIVERGENCE

DELSM(IJ) = D

DEFINE THE CELL AREA AS BEING AN AVERAGE DX TIMES AN AVERAGE DY = CAREA * AVX * AVY, THEN DEFINE AVX(Y) AS THE SQUARE ROOT OF THE AVERAGE OF THE SQUARES OF THE CELL DIAGONAL X(Y)

COMPLEX, THEN, ONE WAY OF DETERMINING AVX(Y) GIVEN AVY(X) IS AVX(Y) = CAREA * ((Y(X)D1**2 + Y(X)D2**2) / 2) ** 0.5 WHERE

Y(X)D1,2 ARE THE CELL DIAGONAL Y(X) COMPONENTS, THUS,

XXA = SQRT((XXA)**2)
YYA = SQRT((YYA)**2)

OKLSM IS 2 * DT * (1, / AVX**2 + 1, / AVY**2) FOR USE IN THE PRESSURE ITERATION

OKLSM(IJ) = 2 * DT * (XXA + YYA) / (XXA + YYA)

XX AND YY ARE THE ACTUAL AVX AND AVY

XX = SQRT(XXA)
YY = SQRT(YYA)
CALCULATE THE EFFECTIVE VISCOSITY COEFFICIENTS, 

LAMBDA(MU) EFFECTIVE = LAM(MU) INPUT TIMES AK WHERE THE LATTER IS DEPENDENT ON THE VALUE OF KXI. IF KXI IS 0 OR 1, AK IS DETERMINED FROM NUMERICAL STABILITY REQUIREMENTS.

KP=KXI+2

GO TO (130, 110, 120), KP

110 CONTINUE

AK=1

GO TO 140

120 CONTINUE

AK=RO(IJ)

GO TO 140

130 CONTINUE

VELIJ=U*UR+V*VR

VELMX=ABS(U4*XX)+ABS(V4*YY)

AK=RO(IJ)+COLAMU*(DT02*VELIJ+VELMX)+EM10

140 CONTINUE

DETERMINE LAMBDA*DIVERSION(VELOCITY)=THIS IS LAMR.

NOTE THAT LAMR IS ALWAYS <= 0.

IF(TURB) ROSIG=RO(IJ)*SIG(IJ)

LAMO=AMIN1(D0,0)*AK*LAM-TWTHRD*ROSIG*O

MUOZ IS HALF MU EFFECTIVE

MUOZ=.5*(AK*MU+ROSIG)

IF(TURB) CAPGAM(IJ)=2.*((DUDR**2+DVDY**2)+.5*(DUDY+DVOR)**2+UOR**2)

PIXX, PIYY, PIXY, AND PITHTA ARE COMPONENTS OF THE VISCOSITY STRESS TENSOR

PIXX=4.*MUO2*DUDR+LAMD

PIYY=4.*MUO2*DVDY+LAMD

PIXY=2.*MUO2*(DUDY+DVOR)

PITHTA=4.*MUO2*UOR+LAMD*CYL

PITH IS 1/4 OF THE CELL AREA * PITHTA

PITH = .25*XY*PITHTA

CALCULATE THE VELOCITY CORRECTIONS FOR THE FOUR CELL VERTICES USING THE EXPLICIT LAGRANGIAN FORM OF THE MOMENTUM EQN.

NOTE THAT THE TILDE VELOCITIES WERE INITIALIZED ABOVE IN BODY FORCES AND THE EFFECTS OF THE

TUGYY=P(IJ)

IF(TURB) TUGYY=TUGYY+TWTHRD*RO(IJ)*TUG(IJ)

YY=Y2+TUGYY

XX=X2+(PIXY*X2+PIXX*Y24)

UTIL(IPJ)=UTIL(IPJ)*DT02H1*(XX*R1*YY=PITH)
SUBROUTINE YPH2

ROUTINE TO DO YAQI PRESSURE ITERATION

ORIGINALLY WRITTEN BY A. A. AMSDEN, LASL T=3

MODIFIED BY J. J. NORTON, LASL T=3, 1975

--- BEGIN COMDECK YSTORE ---

DIMENSION X(1), XPAR(1), R(1), XPAR(1), Y(1), XPAR(1), U(1), UG(1), DELSM(1), V(1), RO(1), R(1), X(1), MPEG(1), RCSO(1), E(1), ETIIL(1), RVOL(1)

LOGICAL RESTRT, FILM, PAPER, TURB

REAL T

COMMON/YSTOR(YSTOR(1), YSCTH(1), YSCTH(1)), A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)

COMMON/YSC2/AAC(1), A0C, A0, A0FAC, A0M, B0, COLAMU, CYL, DR, DT, DTC, DTFAC

COMMON/YSC2/ASC(9600)
COMMON/YSC2/JCEN, JP1, JP2, JP4, JUNF, JUNF2, KX1, LAM, LPB, HU, NAME(8), NCYC, NLC, NPS, NPT, NO1, NO1B, NO12, NRC, NUMIT, ZORIG, OM, OMCYL, PCXCONV, PXL, PXR, PYB, PYCONV, PYT, RDT, REZRON, REZSIE, REZY0, RIBAR, RIBJB, FREZYF, FREZYB, ROMFR, T, THIRD, NCLST, TOUT, TWFIN.

COMMON/YSC2/TC1, TC2, TNEG, TNEGQ, TUSB, TURB, TTOP, PRITE, PBOTH, 1, ILNC, NLC, NPS, NPT, TDBAV, TKB, T1, TUEQENG, EP1, SAV1, OLEVEL, TQ, IST, V, XCONV, XL, XR, YB, YCONV, VT, PTPOLD, DTSL, DTLAST, FYIBQ, IYBQ, YCNVLQ.

3, XCNVLQ, FIXRO, IXRO, IXLO, IYSH, IYSH, QMN, QMX, WMAX, XNM, T2, TLM, 4, ROMFR, ROMFYU, ROMFYB, JDUMP, TWINH, TE, DRY, TMASS, IDTSAV, DTCSAV, IDTV, IDTV, 5, JDTV, IDTC, JDTY, CIRC, TIS, POTE, UMOM, VMDH, TMAG, TGMX, ITM, JTM, ITG, JTG, T, TMASSV, WMAGEF, RMINEF, TSTRTD.

COMMON/YSC2/IZ.

COMMON/YSC4/ITAB(1000).

COMMON/YSC4/ITAB(1000).

* ===== BEGIN COMDECK YAQSC =====

* ===== BEGIN COMDECK YAQEQ =====

EQUIVALENCE(AASC(1), XPAR), (AASC(2), RYPAR), (AASC(3), YMPAR), (AASC(4), U), (AASC(5), V), (AASC(6), RO), (AASC(7), DELM, RCMG, MP), (AASC(8), J), (E, ETIL, AREA, X13K).

*/ 

COMMON/YSC5/RESTRT, FILM, PAPER, IPO, IFD.

* ===== BEGIN COMDECK PARAM =====

COMMON/ZCOM/NSC1, ITAB1, ITABP1, ITABXP1, IPFB1, NP1, NP2, NLC1, NLC2, NLC3, NLC4, IFLMSZ.

1, NLC3, NLC4, IFLMSZ.

* ===== BEGIN COMDECK ASTORE =====

DIMENSION IX(11), IY1(1), IX2(1), XY2(1), XCO(1), YCO(1), CONC1.

EQUIVALENCE(AT, IX1), (AT2, IX2), (AT3, IY1), (AT4, IY2), (AT5, XCO.

* ===== BEGIN COMDECK ASTORE =====

COMMON/EQNST/ROTMP, ETMP, GMONE, CONCJ.

IF THE TURBULENCE IS ON, GO COMPUTE CORRECTIONS DUE TO IT.

IF(TURB) CALL TRBCOR.

CALL START.

DO 20 J=2, JP1.

DO 10 I=1, IBAR.

IMJ=I+Q.

IP=I+J+Q.

IF(JP+IP=Q) GO TO 10.

IF(TURB) CALL TRBCOR.

CALL START.

DO 20 J=2, JP1.

DO 10 I=1, IBAR.

IMJ=I+Q.

IP=I+J+Q.

IF(JP+IP=Q) GO TO 10.

IF(TURB) CALL TRBCOR.
GM1#GMONE
GM1#GM1*(GM1+1.0)

C *** SET DENSITY FOR START OF PRESSURE ITERATION ********

MTIL(IJ)=RO(IJ)*ROL(IJ)*RVOL(IJ)*DT
ROL(IJ)=MTIL(IJ)
RC50(IJ)=1./((EM1O+GM1*AMAX1)*SIE(IJ)*T)
IF(TURB,AND,T,GT,TSTARTD) CONC(IJ)=CONC(IJ)+CTEMP(IJ)*RVOL(IJ)*DT
UG(IJ)=1./(R(IJ)+R(IPJP)+R(IPJP)+R(IPJ))*CYL
X13=1.(IPJ)*X(IPJ)
X13K(IJ)=X13
X24=1.(IPJP)*X(IPJ)
X24K(IJ)=X24
Y13=1.(IPJ)=Y(IPJ)
Y13K(IJ)=Y13
Y24=1.(IPJP)=Y(IPJ)
Y24K(IJ)=Y24
V6(IJ)=1./(X13*Y24*X24*Y13)

XR13K(IJ)=0.5*(R(IPJP)+R(IPJ))*X13
XR24K(IJ)=0.5*(R(IPJP)+R(IPJ))*X24

IF(IPJ,AND,T,GT,TSTARTD) CONC(IJ)=CONC(IJ)+CTEMP(IJ)*RVOL(IJ)*DT
UG(IJ)=1./(R(IJ)+R(IPJP)+R(IPJP)+R(IPJ))*CYL
X13=1.(IPJ)*X(IPJ)
X13K(IJ)=X13
X24=1.(IPJP)*X(IPJ)
X24K(IJ)=X24
Y13=1.(IPJ)=Y(IPJ)
Y13K(IJ)=Y13
Y24=1.(IPJP)=Y(IPJ)
Y24K(IJ)=Y24
V6(IJ)=1./(X13*Y24*X24*Y13)

XR13K(IJ)=0.5*(R(IPJP)+R(IPJ))*X13
XR24K(IJ)=0.5*(R(IPJP)+R(IPJ))*X24

IJP=IPJP

IF(IPJ,AND,T,GT,TSTARTD) CONC(IJ)=CONC(IJ)+CTEMP(IJ)*RVOL(IJ)*DT
UG(IJ)=1./(R(IJ)+R(IPJP)+R(IPJP)+R(IPJ))*CYL
X13=1.(IPJ)*X(IPJ)
X13K(IJ)=X13
X24=1.(IPJP)*X(IPJ)
X24K(IJ)=X24
Y13=1.(IPJ)=Y(IPJ)
Y13K(IJ)=Y13
Y24=1.(IPJP)=Y(IPJ)
Y24K(IJ)=Y24
V6(IJ)=1./(X13*Y24*X24*Y13)

XR13K(IJ)=0.5*(R(IPJP)+R(IPJ))*X13
XR24K(IJ)=0.5*(R(IPJP)+R(IPJ))*X24

IJP=IPJP

IJP=IJP+1.

CALL LOOP

CALL DONE

NUMT=0

MUSTIT=1

PLMAX=EM1O

CALL START

DO 68 J2,JP1
DO 50 T1=18AR
IPJ=IPJ+1.

X1=X(IPJ)
Y1=Y(IPJ)

R1=R(IPJ)
U1=UL(IPJ)

V1=VL(IPJ)

X2=X(IPJP)

Y2=Y(IPJP)

R2=R(IPJP)
U2=UL(IPJP)

V2=VL(IPJP)

X3=X(IPJP)

Y3=Y(IPJP)

R3=R(IPJP)
U3=UL(IPJP)

V3=VL(IPJP)

X4=X(IPJ)

Y4=Y(IPJ)

R4=R(IPJ)
U4=UL(IPJ)

V4=VL(IPJ)

UOR=(U1+U2+U3+U4)*UG(IJ)

RAR*V6(IJ)

DUD=((U1=U3)*(Y2=4)*(U2=U4)*(Y1=Y3))*RAR

DOVY=((V2=V4)*(X1=3)*(V1=V3)*(X2=X4))*RAR

281
SUBROUTINE YPH3

C ROUTINE TO FINISH PHASE 1 AND CALCULATE GRID VELOCITIES
1 NLCP3, NLCP4, IFLMSZ

63 * ===== END COMDECK PARAM =====

64 * ===== BEGIN COMDECK ASTORE =====

65 COMMON/ASTC/AT(100), FT(100)

66 DIMENSION IX1(1), IY1(1), IX2(1), IY2(1), XCO(1), YCO(1), CON(1)

67 EQUIVALENCE(AT, IX1), (AT(2), IX2), (AT(3), IY1), (AT(4), IY2), (AT(5), XCO

68 1 ), (AT(9), YCO), (FT, CON)

69 * ===== END COMDECK ASTORE =====

70 REAL LAMD, MU0

71 DT=1.0E30

72 CALL START

73 DO 40 J=2, JP1

74 DO 30 IM=1, IBAR

75 IMJ=IM+J-1

76 IMJ+J-1

77 JPJ=JP+J-1

78 X1=X(IPJ)

79 Y1=Y(IPJ)

80 R1=R(IPJ)

81 U1=UL(IPJ)

82 V1=VL(IPJ)

83 W1=VL(IPJ)

84 V2=W(IPJ)

85 Y2=Y(IPJ)

86 R2=R(IPJ)

87 U2=UL(IPJ)

88 V2=V(IPJ)

89 X2=X(IPJ)

90 Y2=Y(IPJ)

91 R2=R(IPJ)

92 U2=UL(IPJ)

93 V2=V(IPJ)

94 X2=X(IPJ)

95 Y2=Y(IPJ)

96 R2=R(IPJ)

97 U2=UL(IPJ)

98 V2=V(IPJ)

99 X4=X(IPJ)

100 Y4=Y(IPJ)

101 R4=R(IPJ)

102 U4=UL(IPJ)

103 V4=VL(IPJ)

104 W4=VL(IPJ)

105 V4=VL(IPJ)

106 X31=X3-K(IPJ)

107 Y31=Y3-K(IPJ)

108 Z31=Z3-K(IPJ)

109 X1=X(IPJ)

110 Y1=Y(IPJ)

111 Z1=Z(IPJ)

112 X2=X(IPJ)

113 Y2=Y(IPJ)

114 Z2=Z(IPJ)

115 X3=X(IPJ)

116 Y3=Y(IPJ)

117 Z3=Z(IPJ)

118 HR13=0.5*(R1+R3)
XR13=XR13K(IJ)
XR24=XR24K(IJ)
QX=P(IJ)
RR0=1/RO(IJ)
RMCR=RR0*RVOL(IJ)
QY=QX-PYY
DELE=0.25*DT*RVOL(IJ)*((U1L+U1)*R1+Y24*QX+(U2L+U2)*R2+Y31*QX-(U3L+
1 U3)*R3+Y24*QX=(U4L+U4)*R4+Y31*QX=(V1L+V1)*XR24*QY+(V2L+V2)*XR13*
2 QY=(V3L+V3)*XR24*QY=(V4L+V4)*XR13*QY)
DELE=DELE+0.25*DT*RVOL(IJ)*((U1L+U1)*XX1-PITH)+(U2L+U2)*XX2+PITH
2 )=(U3L+U3)*XX1+PITH)=((U4L+U4)*(XX2+PITH)+(V1L+V1)*XX3+(V2L+V2)*
2 XX4=(V3L+V3)*XX3=(V4L+V4)*XX4
ETIL(IJ)=SIL(IJ)*DELE/RRO
IJ=IPJ
40 CONTINUE
CALL LOOP
40 CONTINUE
CALL DONE
CALL START
IF(NOT,TUR8) GO TO 50
TNEG=TNEG
50 CONTINUE
DO 180 J=2,JP2
DO 120 I=1,IP1
IF(NOT,TUR8) GO TO 60
FF=0.5/RM(IJ)
TK2=TK2+FF*U(IJ)**2+VL(IJ)**2
TP2=TP2+FF*GZ*DT*(VL(IJ)+VL(IJ))
60 CONTINUE
IMJ=IJ+NJ
ITP=IJ+NW
IMJM=IJ+NW
IJM=IJ+NW
VTEMP=1.25/RM(IJ)*AVCU(IJ)*(UTIL(IJ)+U(IJ))+ANCV(IJ)*(VTIL(IJ)+V(IJ)+
1 (IJ)))
XX=1.
YY=1.
IF((I.EQ.1),OR,(I.EQ.IP1)) XX=0.
IF((J.EQ.2),OR,(J.EQ.IP2)) YY=0.
IF((XX*YY).EQ.0.) GO TO 70
E(IJM)=E(IJM)+VTEMP*RVOL(IJM)/RO(IJM)
E(IJM)=E(IJM)+VTEMP*RVOL(IJM)/RO(IJM)
E(IJM)=E(IJM)+VTEMP*RVOL(IJM)/RO(IJM)
E(IJM)=E(IJM)+VTEMP*RVOL(IJM)/RO(IJM)
GO TO 110
70 IF(I.NE.1) GO TO 80
IF((J.EQ.2),OR,(J.EQ.IP2)) GO TO 110
E(IJM)=E(IJM)+2.*VTEMP*RVOL(IJM)/RO(IJM)
E(IJM)=E(IJM)+2.*VTEMP*RVOL(IJM)/RO(IJM)
GO TO 110
80 IF(I.NE.IP1) GO TO 90
IF((J.EQ.2),OR,(J.EQ.IP2)) GO TO 110
E(IMJ) = E(IMJ) + 2 * VTEMP * RVOL(IMJ) / RO(IMJ)
E(IMJ) = E(IMJ) + 2 * VTEMP * RVOL(IMJ) / RO(IMJ)
GO TO 110

90 IF (J EQ 2) GO TO 100
E(IMJ) = E(IMJ) + 2 * VTEMP * RVOL(IMJ) / RO(IMJ)
E(IMJ) = E(IMJ) + 2 * VTEMP * RVOL(IMJ) / RO(IMJ)
GO TO 110

100 E(IJ) = E(IJ) + 2 * VTEMP * RVOL(IJ) / RO(IJ)
E(IMJ) = E(IMJ) + 2 * VTEMP * RVOL(IMJ) / RO(IMJ)

110 CONTINUE
IJ = IJP
IJP = IJP + NQ
IJM = IJ + NQ
CALL LOOP
CONTINUE
IJP = IJP + NQ
CALL DONE
CALL START
IFAKE = IJ
DO 190 J = 2, JP1
DO 180 IPI, IBAR
IPJE = IJP + NQ
IPJP = IJP + NQ
IF (.NOT. TURB) GO TO 100
FF = RO(IJ) / RVOL(IJ)
T2 = T2 + E(IJ) * FF
TQ2 = TQ2 + FF * TQ(IJ)
SS = SS + GZ * DT * GRROZ(IJ)
CONTINUE
IF (J .NE. 2) GO TO 150

150 CONTINUE
IF (J_IE, 2) GO TO 150

****** SET BOTTOM FICTITIOUS ROW **************
ROL(IJM) = RO(IJM)
ETIL(IJM) = ETIL(IJM)
IF (TURB) TUG(IJM) = TUG(IJ)
CONC(IJM) = CONC(IJ)
IJP, NE, IBAR) GO TO 150

****** SET LOWER RIGHT FICTITIOUS CORNER **************
ROL(IJM + NQ) = RO(IJM + NQ)
ETIL(IJM + NQ) = ETIL(IFAKE)
IF (TURB) TUG(IJM + NQ) = TUG(IFAKE)
CONC(IJM + NQ) = CONC(IFAKE)
GO TO 160

160 IF (I, NE, IBAR) GO TO 170

****** SET TOP FICTITIOUS ROW **************
ROL(IJP) = RO(IJP)
IF (TURB) TUG(IJP) = TUG(IJ)
CONC(IJP) = CONC(IJ)

170 IF (I, NE, IBAR) GO TO 170

****** SET RIGHT HAND FICTITIOUS COLUMN **************
ROL(IPJ) = RO(IPJ)
ETIL(IPJ) = ETIL(IPJ)
IF (TURB) TUG(IPJ) = TUG(IJ)
CONC(IPJ) = CONC(IJ)
SUBROUTINE YPLOT

ROUTINE TO DO PLOTTING FOR YAQUI

ORIGINALLY WRITTEN BY A.A.AMSDEN, LASL T-3
MODIFIED AND DOCUMENTED BY J.L., NORTON, LASL T-3, 1975

**** BEGIN COMMON PARAM ****
COMMON/PCOMMON/NCP1,NCP2,ITABP,ITABXP,ITABYP,IPF8,NP1,NP2,NLCP1,NLCP2,
1 NLCP3,NLCP4,IFLMSZ
* ----- END COMDECK PARAM ----- 
12 * ----- BEGIN COMDECK ASTORE ----- 
13 COMMON/ASTC/AT(100),FT(100) 
14 DIMENSION IX1(I),IY1(I),IX2(I),IY2(I),XCO(1),YCO(1),CON(1) 
15 EQUIVALENCE(AT(1),IY1(1),AT(2),IY2(1),AT(3),YCO(1),AT(4),Y2(1),AT(5),XCO 
16 1),XCO(2))/(FT(100)) 
17 * ----- END COMDECK ASTORE ----- 
18 * ----- BEGIN COMDECK ASTORE ----- 
19 EQUIVALENCE(XI(X),YI(Y),R(1),YPAR(1),Y1(I),MPAR(1),U(1),UG(1),DELSM( 
20 1),V(1),VG(1),RO(1),SIE(1),MP(1),RMP(1),RCSQ(1),E(1),ETIL(1),RVOL 
21 2),M(1),R(1),V(1),P(1),PL(1),UP(1),UTIL(1),UL(1),CQ(1),VTIL(1) 
22 3),VL(1),RLC1(R),AVS(1),AVS(1),DLSQ1(I),DLSQ2(I),CAPGM(1),TUG 
23 4),SIG(1),TUS(1),GRO(1),GRDZ(1),GRDZ(1),TUQVEC(1),MTIL(1), 
24 5),CONC(1),CTEMP(1),ANGC(1),ANC(1),GRSV(1),GZSV(1),X13K(1),X24K(1), 
25 6),Y13K(1),Y24K(1),X13K(1),X24K(1),XDKSLM(1),AREA(1) 
26 * ----- END COMDECK YAOSEC ----- 
27 * ----- BEGIN COMDECK YAOSEC ----- 
28 LOGICAL RESTRT,FILM,PAPER,TURB 
29 REAL LAM,MU 
30 COMMON/YSC1/AASC(160) 
31 COMMON/YSC2/AA(1),ANC,AN0,ANFAC,A0M,BO,COLAMU,CY,OR,DT,DT,RFC, 
32 1),DT000,DT010,DT020,DT030,DT040,DT050,DT060,DT070,DT080,DT090,DT100,DT110, 
33 2),FIPY,FIPY,FIXL,FXR,FIY,FIY,FREZXR,GR,GRDVEL,GZ,DEP,I,IBAR, 
34 3),IDT0,1M,14,IM,1IPX,1PXR,IPY,IPY,IP1,ISC2,ISC3,ITV, 
35 4),JUNF,JUX,JUX,YUX,YUX,JBAR, 
36 COMMON/YSC2/IEN,JPT,JP2,JP4,JP6,JP8,JBAR, 
37 1),NCYC,NLC,NPS,NPT,NG,NG1,NG2,NSC,NUMIT,INOR,OM,OMC,PL,PXCONV 
38 2),PLX,PXR,PRY,PYCONV,PM,ROZ,REZRE,REZRE,REZRE,REZRE, 
39 3),FREE,FREE,FREE,ROR,TH,TH OR,THFST,THFIN, 
40 COMMON/YSC2/TQ1,TQ2,TUQ1,TUQ2,TUQ3,TUQ4,TUQ5,TUQ6,TUQ7,TUQ8,TUQ9,TUQ10, 
41 1),INL,NML,NLM,LP,TUQD,TQD1,TQD2,TQD3,TQD4,TQD5,TQD6,TQD7,TQD8, 
42 2),FFX,FXR,FXR,FRX,FRR,FRR,FRR,FRR, 
43 3),XCNVLD,FXLDO,FXLDO,FXLDO, 
44 4),ROFX,ROFX,ROFX,ROFX, 
45 5),JTV,JTV,JTV,JTV,JTV,JTV, 
46 6),TMAS,DMAS,DMAS,DMAS,DMAS,DMAS,DMAS, 
47 7),TMAS,DMAS,DMAS,DMAS,DMAS,DMAS,DMAS, 
48 8),TMAS,DMAS,DMAS,DMAS,DMAS,DMAS,DMAS, 
49 COMMON/YSC2/ZZ 
50 COMMON/YSC4/ITAB(100) 
51 COMMON/YSC4/ITAB(100) 
52 COMMON/YSC5/RESTRT,FILM,PAPER,IPD,IFD 
53 * ----- END COMDECK YAOSEC ----- 
54 * ----- BEGIN COMDECK YAOSEC ----- 
55 EQUIVALENCE(AASC(1),X,XPAR),AASC(2),R,YPAR),AASC(3),Y,MPAR),AASC 
56 1),AASC(2),U),AASC(3),V),AASC(4),RO),AASC(5),RMP),AASC(6),DELSM,RCSQ,MP, 
57 1),E,ETIL,ARE,AR13K, 
58 2),AASC(13),SIE),AASC(14),PMO,DKSLM,RMP),AASC( 
59 3),XVRL),AASC(18),XVRL),AASC(19),XVRL),AASC(20),XVRL), 
60 4),U,PMX,PU),AASC(13),VTEIL,VL,PHV,PU),AASC(14),QCG,POL),AASC(17 
61 5),CAPGM,UG),AASC(18),TUG),AASC(19),SIG),AASC(20),TUS),AASC( 
62 6),GRDZ),AASC(22),GRDZ),AASC(23),DLSQ1,I13K),AASC(24),GZSV 
63 7),AASC(25),DLSQ2,VG),AASC(26),GZSV),AASC(27),GRDZ,GRDZ, 
64 8),Y24K),AASC(28),MTIL),AASC(29),CONC),AASC(30),CTEMP,XT24K), 
65 9),AASC(31),ANGC),AASC(32),ANGC),AASC(33),AVSVX13K),AASC(34), 
66 1),AVSVX24K) 
67 REAL LAM,MU,MPAR,MTIL
* \text{END COMDECK} VAGEO \text{END COMDECK YSTORE} *

* \text{BEGIN COMDECK PCALL} *

\text{COMMON/PCALL/XCONVP,YCONVP,YUP,YLB} *

* \text{END COMDECK PCALL} *

\text{LOGICAL EULER}

\text{DIMENSION BCD(2)}

\text{DIMENSION ITITLE(6,8), INWT(8)}

\text{DIMENSION IJL1(6), IJL2(6), IJL3(6), IJL4(6), IJL5(6), IJL6(6)} \text{DIMENSION IJL7(6), IJL8(6)}

\text{EQUIVALENCE(ITITLE, IJL1), (ITITLE(7), IJL2), (ITITLE(13), IJL3),} \text{ITITLE(19), IJL4), (ITITLE(25), IJL5), (ITITLE(31), IJL6), (ITITLE(37),}

2 IJL7), (ITITLE(43), IJL8)

\text{DATA IJL1/7HDENSITV, 5*1H/}

\text{DATA IJL2/24HSPECIFIC INTERNAL ENERGY, 3*1H/}

\text{DATA IJL3/9H VORTICITY, 5*1H/}

\text{DATA IJL4/18H VELOCITY MAGNITUDE, 6*1H/}

\text{DATA IJL5/26H SPECIFIC TURBULENCE ENERGY, 3*1H/}

\text{DATA IJL6/30H RATIO OF SPEC, TURB, ENERGY TO,}

\text{DATA IJL7/21H SPEC, KINETIC ENERGY/}

\text{DATA IJL7/26H TURBULENCE VISCOITY (SIGMA), 3*1H/}

\text{DATA IJL8/22H CONCENTRATION X RADIUS, 3*1H/}

\text{DATA INWT/1, 3, 1, 2, 3, 6, 3, 3/}

\text{DATA BCD/1H/}

\text{DATA VMAX, DRMIN, DZMIN, DRMAX, DZMAX/5*0.,/}

\text{DATA TP/0.,/}

\text{CALL SECONDS(TP)}

\text{WRITE(59, 210) TP}

\text{IF THERE ARE ANY PARTICLES, GO PLOT THEM}

\text{IF(NPT, GT, @) CALL PARPLT}

\text{EULER IS TRUE IF THE CALCULATION IS EULERIAN}

\text{EULER = .FALSE.,}

\text{IF(GROVEL, EQ, @) EULER = .TRUE.,}

\text{INITIALIZE COORDINATE-TO-RASTER CONVERSION FACTORS IN COMMON}

\text{YCONVP = YCONV}

\text{XCONVP = XCONV}

\text{INITIALIZE MINIMUM PLOT Y VALUE IN COMMON}

\text{YLB = YB}

\text{GO GET THE MAXIMUM VELOCITY ABSOLUTE VALUE ALONG THE COORDINATE}

\text{DIRECTIONS AND THE MINIMUM AND MAXIMUM ZONE SIZES}

\text{CALL DVMM(VMAX, IVM, JVM, DRMIN, DZMIN, DRMAX, DZMAX)}

\text{VTMAX = 0.5*VMAX}

\text{IF DOING AN EULERIAN CALCULATION, ONLY DO ZONE PLOTS UPON}

\text{STARTUP, OTHERWISE, DO THEM EVERY OUTPUT CYCLE,}

\text{IF(EULER, AND, NCYC, NE, @) GO TO 20}
CALL ADV(1)
CALL ZONPLT

PUT LABELS ON THE ZONE PLOT AND OUT TO THE CYCLE SUMMARY PRINT

CALL LINCNT(59)
WRITE(IFD,220)
DO 10 IPX=1,IFD,6
10 WRITE(IPX,250) ORMIN,DRMAX,DRMIN,DRMAX,XR,YB,YT
WRITE(IFD,260) JNM,NAME,T,NCYC
20 CONTINUE

SKIP VELOCITY PLOTS IF VELOCITIES ARE ESSENTIALLY ZERO,
OTHERWISE, PLOT VELOCITIES SCALLED TO THE MAXIMUM VELOCITY,

IF(VMAX,GT.EM10) CALL VELPLT(VMAX,0)

THE FULL PLOTS HAVE BEEN MADE. ADJUST PLOTS NOW SO THAT THE
REGION DISPLAYED INCLUDES THAT PART OF THE PROBLEM TWO
FIREBALL RADII ABOVE THE TOP OF THE FIREBALL TO THREE RADII
BETWEEN THE BOTTOM OF THE FIREBALL

YUP=PTOP+2.*PR17E
YLBR=BOTM-3.*PR17E
IF(YUP,GT,YT) YUP=YT
IF(YLB,LT,YB) YLB=YB

READJUST THE CARTESIAN TO RASTER RATIOS TO ENCOMPASS THE NEW
REGION BUT SAVE THE OLD COORDINATES

FIY80=FIYB
IYB=IYB
FIY8=900
IYB=900
YCONVP=FLOAT(YT-IYB)/(YUP-YLB)
XCONVP=XCONV*YCONVP/YCONV
FIXLO=FIXL
IXLO=IXL
FIXRO=FIXR
IXRO=IXR
FIXL=65,
IXL=65
FIXR=660,
IXR=660
YCNVLD=YCONV
YCONV=YCONVP
XCNVLD=XCONV
XCONV=XCONVP

DO THE PLOTS AGAIN FOR THE SMALLER REGION

IF(EULER,.AND.,NCYC,NE,0) GO TO 30
CALL ADV(1)
CALL ZONPLT

PUT LABELS ON THE ZONE PLOT
CALL LINCNT(59)
WRITE(IFD,230)
WRITE(IFD,250) DRMIN,DRMAX,DZMIN,DZMAX,XR,YB,YT
WRITE(IFD,260) JNM,NAME,T,NEYC

DRAW THE PLOTTING RECTANGLE AND LABEL IT
CALL TICBOX
CONTINUE

SKIP VELOCITY PLOTS IF VELOCITIES ARE ESSENTIALLY ZERO
IF(VMAX,LT,EM10) GO TO 40
PLOT VELOCITIES SCALED TO MAXIMUM VELOCITY
CALL VELPLT(VMAX,0)
CALL TICBOX
PLOT UNSCALED VELOCITIES
CALL VELPLT(VMAX,1)
CALL TICBOX
CONTINUE

SECTION TO DO CONTOUR PLOTS
L=1, DENSITY (ISOPYCNICS)
L=2, SPECIFIC INTERNAL ENERGY (ISOTHERMS)
L=3, VORTICITY
L=4, VELOCITY MAGNITUDE
L=5, SPECIFIC TURBULENCE ENERGY
L=6, RATIO OF SPEC. TURB. ENERGY TO SPEC. KINETIC ENERGY
L=7, TURBULENCE VISCOSITY (SIGMA)
L=8, CONCENTRATION X RADIUS
IF TURBULENCE IS NOT ON, DO NOT PLOT THE TURBULENCE QUANTITIES
LP#8
IF(.NOT.,TURB) LP#4
PREPARE THE PLOTTING ARRAYS
DO 200 L=1,LP
CHECK FOR VORTICITY PLOT
IF(L,NE,3) GO TO 50
YES, SPECIAL SUBROUTINE PREPARES VORTICITY,
CALL GETOMG
GO TO 190
CONTINUE
CALL START
DO 180 J=2,JP1
DO 170 I=1,IBAR
GO TO (60,70,200,80,80,140,80), L
60 CONTINUE
C
DENSITY
C
CQ(IJ)=RQ(IJ)
GO TO 160
70 CONTINUE
C
SPECIFIC INTERNAL ENERGY
C
CQ(IJ)=SIE(IJ)
GO TO 160
80 CONTINUE
C
IPJ*IJ+NQ
IPJP=IJP+NQ
IF(L.EQ.5.0R.LOE$I,8,0)
XXA=(0.25*(U(IJ)+U(IPJ)+U(IPJP)+U(IJP)))*2+(0.25*(V(IJ)+V(IPJ)+V(IJP)))*2
GO TO 100
90 CONTINUE
RAV=0.25*(R(IJ)+R(IPJ)+R(IPJP)+R(IJP))
100 CONTINUE
LPX=L+3
GO TO (110,120,130,140,150),LPX
110 CONTINUE
C
VELOCITY MAGNITUDE
C
CQ(IJ)=SQRT(XXA)
120 CONTINUE
C
SPECIFIC TURBULENCE ENERGY
C
CQ(IJ)=TUE(IJ)
130 CONTINUE
C
RATIO OF SPEC, TURB, ENERGY TO SPEC, KINETIC ENERGY
C
CQ(IJ)=TUE(IJ)/(XXA+EM10)*2,
TEST=AMAX1(ABS(U(IJ)),ABS(V(IJ)))
IF(TEST,LT,VTEST) CQ(IJ)=0,
GO TO 160
140 CONTINUE
C
TURBULENCE VISCOITY (SIGMA)
C
CQ(IJ)=SIG(IJ)
GO TO 160
150 CONTINUE
C
CONCENTRATION X RADIUS
C
CQ(IJ)=CONC(IJ)*RAY
160 CONTINUE
IF(ABS(CQ(IJ)),LT,1.E-50) CO(IJ)=0,
      IJP=IJP+NO
170 IJ=IJ+NO
CALL LOOP
180 CONTINUE
CALL DONE
190 CONTINUE
C DECIDE WHETHER TO REQUEST LOGARITHMIC OR LINEAR CONTOUR INCREMENTS
304 C PLOTS 1 AND 2 ARE LOGARITHMIC = THE OTHERS LINEAR
307 C
308   ILOG=1
309   IF(L.GT,2) ILOG=0
310 C GO DO THE PLOT
312 C
313 C ITITLE IS THE PLOT TITLE
314 C INWT IS THE NO. OF COMPUTER WORDS IN THE TITLE
315 C
316   CALL CONTOF(ILOG,ITITLE(1,L),INWT(L)))
317 290 CONTINUE
318 C
319 C RESTORE THE OLD RASTER VALUES
320 C
321   FIYB=FIYBO
322   IYB=IYBO
323   YCONV=YCNVLD
324   XCONV=XCNVLD
325   FIXR=FIXRO
326   FIXL=FIXLO
327   IXR=IXRO
328   IXL=IXLO
329   CALL SECOND(TP)
330   WRITE(59,240) TP
331   RETURN
332 C
333   250 FORMAT(1H ,20HBEGIN YPLOT AT CP = ,F10.4)
334   220 FORMAT(10H ALL ZONES)
335   230 FORMAT(29H ZONES IN THE FIREBALL REGION)
336   240 FORMAT(1H ,18HENDE YPLOT AT CP = ,F10.4)
337   250 FORMAT(7H ORMIN=1PE12,5,7H ORMAX=E12,5,7H DRMIN=E12,5/7H DRMAX=E12
338   1 .5,4H XR=E12,5,4H YB=E12,5,4H YT=E12,5)
339   260 FORMAT(1H ,9XAI0,8A10,3X2MT#,1PE12,5,1X6HCYCLE#,IS)
340 END

--------------------------------------------------------------------------------

1 C SUBROUTINE ZONPLT
2 C ROUTINE TO DO FULL AND PARTIAL ZONE PLOTS
4 C ORIGINALLY WRITTEN BY A, A, AMSDEN, LASL T=3
**MODIFIED AND DOCUMENTED BY J.L. NORTON, LASL T-3, 1975**

---

**BEGIN COMDECK PARAM**

- COMMON/PATOM/NSCPI, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
- NLCP3, NLCP4, FIRMZ,

**BEGIN COMDECK YSTORE**

- COMMON/PCOM/NSCP1, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
- NLCP3, NLCP4, FIRMZ,

**BEGIN COMDECK YAGDM**

- COMMON/PATOM/NSCPI, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
- NLCP3, NLCP4, FIRMZ,

**BEGIN COMDECK YAGDT**

- COMMON/PATOM/NSCPI, ITABP, ITABXP, ITABYP, IPFB, NP1, NP2, NLCP1, NLCP2,
- NLCP3, NLCP4, FIRMZ,
**END COMDECK YSTORE**

**BEGIN COMDECK PCALL**

```plaintext
COMMON/PCALLC/XCONVP,YCONVP,YUP,YLB
```

**END COMDECK PCALL**

```
LOOP OVER ALL REAL ZONES
```

```plaintext
JA = 0
CALL START
DO 60 J = 2, JP1
DO 50 I = 1, IBAR
IPJ = I + J
IPJP = I + J + 1
DETERMINE THE FOUR VERTICES OF CELL (I, J)
```

```plaintext
(X1, Y1) IS VERTEX (I+1, J) (VERTEX 1)
(X2, Y2) IS VERTEX (I+1, J+1) (VERTEX 2)
(X3, Y3) IS VERTEX (I, J+1) (VERTEX 3)
(X4, Y4) IS VERTEX (I, J) (VERTEX 4)
```

```plaintext
X1 = X(IPJ)
X2 = X(IPJP)
X3 = X(IJP)
Y1 = Y(IPJ)
Y2 = Y(IPJP)
Y3 = Y(IJP)
Y4 = Y(IJ)
```

```plaintext
(ixn, iyn) ARE THE PLOT RASTER COORDINATES OF VERTEX N
```

```
IX1 = FIXL + (X1 - XL) * XCONVP
IX2 = FIXL + (X2 - XL) * XCONVP
IX3 = FIXL + (X3 - XL) * XCONVP
IX4 = FIXL + (X4 - XL) * XCONVP
```

```plaintext
IV1 = FYB + (Y1 - YLB) * YCONVP
IV2 = FYB + (Y2 - YLB) * YCONVP
IV3 = FYB + (Y3 - YLB) * YCONVP
IV4 = FYB + (Y4 - YLB) * YCONVP
```

```
IF VERTEX 1 OR 2 IS ABOVE THE TOP OR BELOW THE BOTTOM OF THE PLOT REGION, DO NOT PLOT THIS ZONE
```

```
IF (IY1, GT, IY3, OR, IY1, LT, IYT) GO TO 40
```

```
IF (IY2, GT, IY3, OR, IY2, LT, IYT) GO TO 40
```

```
IF VERTEX 1 OR 2 IS TO THE RIGHT OF THE PLOT REGION, SKIP THIS ZONE
```

```
IF (IX1, GT, IXR) GO TO 40
```

```
IF (IX2, GT, IXR) GO TO 40
```

```
IF THIS ZONE IS ON THE SYMMETRY AXIS, DRAW THE LEFT SIDE
```

```
IF (I, EQ, 1) CALL DRV(IX3, IY3, IX4, IY4)
```

```
IF THIS ZONE IS IN THE BOTTOMMOST ROW TO BE PLOTTED, DRAW THE
```

296
120 C BOTTOM SIDE
121 C
122 IF(JB,NE,0) GO TO 10
123 JB=
124 GO TO 20
125 10 CONTINUE
126 IF(J,NE,JB) GO TO 30
127 20 CONTINUE
128 CALL DRV(IX4, IY4, IX1, IY1)
129 30 CONTINUE
130 C
131 C DRAW THE RIGHT SIDE
132 C
133 CALL DRV(IX1, IY1, IX2, IY2)
134 C DRAW THE TOP SIDE
135 C
136 CALL DRV(IX2, IY2, IX3, IY3)
137 40 IJ=IPJ
138 50 IJP=1PJP
139 CALL LOOP
140 60 CONTINUE
141 RETURN
142 END

REFERENCES