Lawrence Livermore Laboratory

ABMAC-Arbitrary Boundary Marker and Cell Eulerian Hydrodynamic Incompressible Numerical Method

W. J. Garcia, Jr. and J. A. Viecelli

June 1976

NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE. It has been reproduced from the best available copy to permit the broadest possible availability.

MASTER

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the laboratory.

Prepared for U.S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

DISCLAIMER

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

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

(M)

ABMAC-Arbitrary Boundary Marker and Cell Eulerian
Hydrodynamic Incompressible Numerical Method

by

W. J. Garcia, Jr. *

and

J. A. Viecelli

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

ABSTRACT

This report is intended to be a "user manual" for the Lawrence Livermore Laboratory version of the Eulerian incompressible hydrodynamic computer code ABMAC. The theory of the numerical model is discussed in general terms. The format for data input, and data printout is described in detail. A listing and flow chart of the computer code are provided.

*Permanent address: Bechtel Corporation, 50 Beale Street, San Francisco, California.

TABLE OF CONTENTS

Abstract

Introduction

The Numerical Model

The Computer Program

Bibliography

Appendix A Definition of Symbols

B Input Card Format for DATA

C Output Edit Format

D Fortran Listing

E Flowchart Diagrams

INTRODUCTION

This report is intended to be a "user manual" for the marker and cell computer code ABMAC, and as such deals primarily with the input of the data, the computer program and printout. The theory of the marker and cell method are described in detail by Welch, Harlow, Shannon and Daly (1966). The extension of the method to incorporate arbitrarily shaped moving boundaries was reported in detail by Vieceilli (1971).

The format for the input data and the meaning of each input parameter is described. All variables listed in the COMMON of the program are defined. The format of the data printout is also discussed in detail and examples of the data printout are included.

A complete listing of the FORTRAN statements and a flow chart of the computer code is included to assist the user in trouble-shooting problems and to assist in making modifications.

For further information on the theory of the method, and discussions of examples of its use and also its limitations, an extensive bibliography of reports and articles specifically applicable to the marker and cell method is included.

THE NUMERICAL MODEL

The Eulerian incompressible hydrodynamic numerical modeling method was originally developed at the Los Alamos Scientific Laboratory (Fromm 1963, Welch, Harlow, Shannon and Daly 1965) and modified by Vieceilli (1969 and 1971) at the Lawrence Livermore Laboratory. The method, designated ABMAC for arbitrary boundary marker and cell method, makes use of finite-difference approximations to the full Navier-stokes equations and continuity equation.

The utility of the method has been demonstrated in numerous applications and has been extended considerably by the modifications made by Vieceilli in extending the method to allow arbitrarily shape moving boundaries. In its original form, boundaries were required to coincide with the computational mesh lines. In Vieceilli's method, the arbitrarily shaped boundaries are treated as free surfaces with an external pressure distribution which causes this "pseudo" free surface to coincide with the specified boundary. Additional modifications were made to improve the resolution of the free surface so that relatively small surface motions can be accurately measured.

The differential equations which are to be approximated by finite difference equations are the continuity equation

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \quad (1)$$

and the Navier-Stokes equations,

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} + \frac{\partial p}{\rho \partial x} = \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + a_x \quad , \quad (2)$$

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \frac{\partial p}{\rho \partial y} = \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + a_y \quad . \quad (3)$$

The Navier-Stokes equations are written in the above form because the finite difference approximation of this form retains rigorous conservation of momentum.

The equations are in two-dimensional Cartesian coordinates, where x and y are the horizontal and vertical coordinates respectively with the origin located at the lower left. The corresponding components of velocity are u and v . The pressure is p , the constant density is ρ and the kinematic viscosity is ν . The horizontal and vertical components of the constant body acceleration are a_x and a_y respectively.

The finite-difference approximation for the Navier-Stokes equations can be written in the following form, with the time derivatives approximated by simple forward differences and the spatial derivatives by centered differences. The pressure is located at the center of the computational cell (k, ℓ) , the horizontal component of velocity u is centered at $(k+1/2, \ell)$ and the vertical component of velocity is centered at $(k, \ell+1/2)$. The velocity component u at advanced time step $n+1$ is

$$\begin{aligned}
 u_{k+1/2, \ell}^{n+1} = & u_{k+1/2, \ell}^n - \frac{\Delta t}{\Delta x} \left[(u^2)_{k+1, \ell}^n - (u^2)_{k, \ell}^n \right] - \frac{\Delta t}{\rho \Delta x} \left[p_{k+1, \ell} - p_{k, \ell} \right] \\
 & - \frac{\Delta t}{\Delta y} \left[(uv)_{k+1/2, \ell+1/2}^n - (uv)_{k+1/2, \ell-1/2}^n \right] - a_x \Delta t \\
 & + \frac{\nu \Delta t}{(\Delta y)^2} \left[u_{k+1/2, \ell+1}^n - 2u_{k+1/2, \ell}^n + u_{k+1/2, \ell-1}^n \right] \\
 & - \frac{\nu \Delta t}{\Delta x \Delta y} \left[v_{k+1, \ell+1/2}^n - v_{k+1, \ell-1/2}^n - v_{k, \ell+1/2}^n + v_{k, \ell-1/2}^n \right] \quad (4)
 \end{aligned}$$

which can be rewritten as

$$u_{k+1/2,l}^{n+1} = u_{k+1/2,l}^n - \frac{\Delta t}{\rho \Delta x} \left[p_{k+1,l} - p_{k,l} \right] \quad (5)$$

where $u_{k+1/2,l}^n$ is the old velocity, advection, body force and viscous terms lumped together. Similarly the vertical velocity v at advanced time step $n+1$ is

$$\begin{aligned} v_{k,l+1/2}^{n+1} = & v_{k,l+1/2}^n - \frac{\Delta t}{\Delta z} \left[(v^2)_{k,l+1}^n - (v^2)_{k,l}^n \right] - \frac{\Delta t}{\rho \Delta y} \left[p_{k,l+1} - p_{k,l} \right] \\ & - \frac{\Delta t}{\Delta x} \left[(uv)_{k+1/2,l+1/2}^n - (uv)_{k-1/2,l+1/2}^n \right] - a_y \Delta t \\ & - \frac{v \Delta t}{\Delta x \Delta y} \left[u_{k+1/2,l+1}^n - u_{k+1/2,l}^n - u_{k-1/2,l+1}^n + u_{k-1/2,l}^n \right] \\ & + \frac{v \Delta t}{(\Delta x)^2} \left[v_{k-1,l+1/2}^n - 2v_{k,l+1/2}^n + v_{k+1,l+1/2}^n \right] \end{aligned} \quad (6)$$

which can be rewritten as

$$v_{k,l+1/2}^{n+1} = \xi_{k,l+1/2}^n - \frac{\Delta t}{\rho \Delta y} \left[p_{k,l+1} - p_{k,l} \right] \quad (7)$$

The solution obtained for the pressure $p_{k,l}$ must be such that the advance time velocity components $u_{k+1/2,l}^{n+1}$ and $v_{k,l+1/2}^{n+1}$ satisfy the continuity equation

$$(\nabla \cdot \mathbf{V})_{k,l}^{n+1} = \frac{1}{\Delta x} \left[u_{k+1/2,l}^{n+1} - u_{k-1/2,l}^{n+1} \right] + \frac{1}{\Delta y} \left[v_{k,l+1/2}^{n+1} - v_{k,l-1/2}^{n+1} \right] = 0 \quad (8)$$

The pressure and velocity are solved simultaneously using the iteration equation for the $i+1$ iteration value of the pressure

$$p_{k,l}^{i+1} = p_{k,l}^i - \Delta \tau (\nabla \cdot \mathbf{V})_{k,l}^{n+1,i} \quad (9)$$

where $\Delta \tau$ is a relaxation parameter. After substituting in the continuity equation one obtains

$$p_{k,l}^{i+1} = p_{k,l}^i + \frac{\Delta\tau\Delta t}{\rho} \left[\frac{p_{k+1,l}^i - 2p_{k,l}^i + p_{k-1,l}^i}{(\Delta x)^2} + \frac{p_{k,l+1}^i - 2p_{k,l}^i + p_{k,l-1}^i}{(\Delta y)^2} \right] - \Delta\tau S_{k,l} \quad (10)$$

where $S_{k,l}$ is a source term of the parameters $\eta_{k+1/2,l}^n$ and $\xi_{k,l+1/2}^n$ which are evaluated only once at each time step. The numerical stability requirement for this iteration equation is that $4\Delta\tau\Delta t/\rho(\Delta x)^2 \leq 1$ and $4\Delta\tau\Delta t/\rho(\Delta y)^2 \leq 1$.

It can be seen that the above finite-difference equation for the iteration of the pressure $p_{k,l}$ is a finite-difference approximation of the equation

$$\frac{\partial p}{\partial \tau} = \frac{\Delta t}{\rho} \nabla^2 p - S \quad (11)$$

In proceeding with the iteration, the asymptotic solution of that equation is approached, which is

$$\nabla^2 p = \rho \frac{S}{\Delta t} \quad (12)$$

A much improved iteration formula can be obtained by replacing $p_{k-1,l}^i$ and $p_{k,l-1}^i$ with the latest values $p_{k-1,l}^{i+1}$ and $p_{k,l-1}^{i+1}$ respectively as one progresses through the mesh in the direction of increasing k and l . Thus

$$p_{k,l}^{i+1} = p_{k,l}^i + \frac{\Delta\tau\Delta t}{\rho} \left[\frac{p_{k+1,l}^i - 2p_{k,l}^i + p_{k-1,l}^{i+1}}{(\Delta x)^2} + \frac{p_{k,l+1}^i - 2p_{k,l}^i + p_{k,l-1}^{i+1}}{(\Delta y)^2} \right] - \Delta\tau S_{k,l} \quad (13)$$

This equation has the advantage of remaining stable with a larger time step Δt , and it will approach steady state more quickly. In this later equation $2\Delta\tau\Delta t/\rho(\Delta x)^2 \leq 1$ and $2\Delta\tau\Delta t/\rho(\Delta y)^2 \leq 1$ are required for stability. In order to use the above equation the values of

$$\left(u_{k+1/2,\ell}^{n+1}\right)^{i+1}, \left(u_{k-1/2,\ell}^{n+1}\right)^{i+1}, \left(v_{k,\ell+1/2}^{n+1}\right)^{i+1}, \text{ and } \left(v_{k,\ell-1/2}^{n+1}\right)^{i+1}$$

must be computed from Eqs. (5) and (7) immediately after the value of $p_{k,\ell}^{i+1}$ is obtained for a mesh cell before going to the next cell.

The arbitrary boundaries are specified by the points of intersection of the boundary lines with the mesh lines. Consequently the arbitrary boundaries are treated as a series of straight lines across the computational cells. Since the arbitrary boundary is determined by only two intersection points in each cell, the Eulerian mesh must be fine enough to resolve sufficiently the shape of the boundaries. The cells containing the arbitrary boundaries are flagged and the coordinates of the midpoint of the boundary segment in each cell and the direction of the normal to the boundary segment in each cell are calculated. The convention is established that the boundary normal always points into the fluid as one progresses along the boundary with the fluid on the left. With this information a pressure distribution at the boundary can be established so that the fluid moves along the boundary and not across it. In the pressure iteration for the cells containing a segment of the arbitrary boundaries, the divergence of the velocity is replaced by the dot product of the normal defining the boundary segment and the velocity of the fluid relative to the boundary. This quantity represents a flux across the boundary, and is analogous to the divergence. Thus for the boundary cells the pressure iteration equation is

$$p_{k,\ell}^{i+1} = p_{k,\ell}^i - \frac{\Delta t}{\Delta x} \{[(\vec{V}_p^{n+1})^i - \vec{V}_b(\vec{r},t)] \cdot \vec{n}\}_{k,\ell} \quad (14)$$

where $(\vec{V}_p^{n+1})^i$ is the velocity vector of the fluid at the midpoint of the boundary segment, $\vec{V}_b(\vec{r},t)$ is the velocity vector of the midpoint of the

boundary segment and n is the normal vector defining the boundary segment associated with cell (k,l) . Here $\Delta\tau$ is the relaxation parameter and Δx is the cell width assuming $\Delta x = \Delta y$. A detailed discussion of this numerical modeling method is given by Vieceilli (1971).

The location of the free surface can be specified by putting marker particles in the cells which contain fluid at the start of the calculations. The computational cells are flagged full, surface or empty depending on whether they and the adjacent cells contain particles. Surface cells are those which contain fluid and are adjacent to an empty cell. The marker particles serve no function other than to specify which cells contain fluid. The fluid density is specified separately in the calculations; it is constant and independent of the number of marker particles in each cell. There must, however, be sufficient marker particles such that no cells become devoid of particles when the physics of the problem indicates that they should not be. The marker particles are moved once each computational cycle after the Poisson equation for the pressure is solved for the entire mesh. The marker particles are moved with an averaged local velocity and the full, surface and empty cell flags are reset by examining the new coordinates of the particles. The free surface is the locus of cell centers of the surface cells. Improved accuracy can be obtained by defining an array of special surface particles which locate the free surface more precisely. This modification was made in order to obtain the accuracy required to solve problems with small surface displacements such as in tsunami problems. The surface particles are moved with the fluid as are the interior marker particles. The pressure at the center of the surface cells is interpolated between the ambient

pressure at the free surface and the value of the pressure in the adjacent full cell. Without the special surface particles the pressure at the center of the surface cells is set equal to the ambient pressure.

In the cells containing arbitrary boundaries a similar situation exists, it is insufficient to say that the cell contains fluid, and the location of the particles must be known relative to the arbitrary boundary. The fluid is assumed to be in contact with the boundary if the particles are within a fraction ϵ of the cell width Δx from the boundary

$$(\vec{X}_p - \vec{X}_n) \cdot \vec{n} < \epsilon \Delta x \quad (15)$$

where $\vec{X}_p$ is the particle position, $\vec{X}_n$ is the position of the midpoint of the boundary segment and n is the normal vector associated with the boundary segment.

A special problem exists in the cells containing both an arbitrary boundary and a free surface. The pressure should be equal to the ambient pressure; however, sometimes it is not possible to satisfy simultaneously the conditions of zero flux across the boundary, vanishing divergence and ambient pressure in the surface cell. This happens when the fluid moves along the boundary in a thin stream or jet which is too small to be resolved by the Eulerian mesh. In this case the pressure is set equal to the ambient pressure and the velocity is adjusted so that the flux across the boundary is zero. The result in this case is that the intersection of the boundary and free surface is not resolved any finer than the cell dimensions. This sometimes results in extraneous particle motion at the boundary and an error

in the location of the free surface of as much one cell width. This error is evident on the plots of the surface profiles which compare the experimental and numerical results, and is further discussed in that section.

In addition to the restrictions on the parameters $\Delta\tau$ and Δt for the numerical stability of the pressure iteration Eq. (13), there are also three further restrictions on the time step Δt . The first restriction results from the finite-difference approximation of the viscous diffusion terms in the momentum Eqs. (4) and (6). The restriction is that

$$\Delta t \leq (\Delta x)^2 / \Delta \nu \quad (16)$$

where Δx is the smallest mesh size and ν is the kinematic viscosity. In order to prevent particles from moving outside the boundaries

$$\Delta t \leq \frac{\epsilon \Delta}{|\vec{V}|_{\max}} \quad (17)$$

where ϵ is the boundary-sensing parameter from Eq. (15) and $|\vec{V}|_{\max}$ is the maximum magnitude of the velocity.

The third restriction is a combination restriction on Δt and the viscosity ν when using centered differencing for advection terms. A truncation error analysis by Hirt (1968) shows that the addition of a small amount of viscosity will dump out instabilities that would otherwise develop. The viscosity should be

$$\nu > \frac{\Delta t |\vec{V}|_{\max}^2}{2} + \frac{\Delta x^2}{2} \left| \frac{\partial \mu}{\partial x} \right|_{\max} \quad (18)$$

By using backward differencing for the advection terms this restriction is removed and Δt must not exceed $\Delta x / |\vec{V}|_{\max}$ to prevent particles from moving across two cell boundaries in one cycle.

The calculations proceed as follows after the initial conditions are generated at time step $n = 0$ and time $t = 0$. The parameters $\eta_{k+1/2, \ell}^n$ and $\xi_{k, \ell+1/2}^n$ are calculated by taking the sum of the advection, viscous diffusion and gravitational acceleration terms in the momentum equations and the old velocities. The pressure field and advanced time velocity components are solved for using the simultaneous iteration technique. Next, the interior marker particles and surface particles are moved. The points defining the arbitrary boundaries are moved and the new boundary normals are generated. The boundary cell flags and the free surface cell flags are reset, the time is incremented by Δt and the next cycle $n+1$ is ready to begin. Also at this point desired data may be edited. This process then continues until the solution to the problem has proceeded as far as is of interest.

This numerical model has been verified with both analytical solutions and laboratory experiments. The comparison with analytical solutions were made with the steady state solution for a jet impinging on a flat plate and on a wedge. In both cases the numerical calculated and theoretical pressure and surface configuration were compared, the agreement was excellent (Viecelli, 1971). In order to verify that the numerical model would faithfully reproduce nature, a series of experiments were run in which water waves were generated by the single impulse motion of vertical and slope walls and an underwater step. In these comparisons, water surface profiles measured from movies taken of the laboratory experiments were compared with the water surface profiles produced by the numerical model calculations. The numerical model was designed to reproduce the laboratory model as exactly as possible. In these cases there was also excellent agreement between the laboratory experiments and the numerical model (Garcia, 1972).

THE COMPUTER PROGRAM

In this section the computational cycle and general description of each subroutine is discussed. The computational cycle can be divided into two relatively separate sections, plus the generation of initial conditions. The first item in the sequence of running a problem is the generation of the initial conditions from the input data. The format of the input data card deck is described in Appendix B. After the initial conditions are established the code cycles through the computational sequence. The first section of the computational sequence is the calculation of the advection, body force and viscous terms from the newly calculated velocity. The terms horizontal and vertical components of the sum of these are stored in the variables ETA and PSI. The second section in the computational cycle is the solution of the pressure iteration equation in which the advanced time solution for the pressure and velocity in each cell is calculated. Upon completion of the iteration of the pressure and velocity, the time step is advanced, the bookkeeping such as moving particles and boundaries, and re-setting flags is accomplished; and ETA and PSI are calculated again for the next cycle. Fig. 1 shows a diagram of the computational cycle.

The subroutine MAIN contains the variables stored in common and the packing functions for the packed variables.

The subroutine MACMAIN is the principal subroutine in the computational cycle. MACMAIN calls subroutine GEN to read in the data and set up the arrays for the start of a new problem or calls RDX to read a tape for re-starting a problem. The advection terms, and viscous terms are calculated

for each cell and added to the old velocity and body force terms to form the variables ETA (KM) and PSI (KM) which correspond to $\eta_{k+1/2,l}^n$ of Eq. (5) and $\xi_{k,l+1/2}^n$ of Eq. (7) respectively. These values remain unchanged for the remainder of the computational cycle. The advection terms in ETA and PSI can be calculated by either centered differencing or backward differencing depending on whether the value of NGEN on optional data card 2 is 1 or 0 respectively. Upon completion of the calculation of ETA and PSI for each cell the advanced time velocities and pressure are calculated, subroutines PRES is called for the calculations. Subroutine VEL57 is called to calculate new velocities at the intersection of free surfaces and arbitrary boundaries. Subroutine TRAJ is called to move the marker particles and free surface particles. Subroutine FLOWIN is called to input particles at the flow-in boundaries. Subroutine SHIFT is called to move the arbitrary Lagrangian boundaries and recalculate boundary position and boundary normals. Subroutine REFLAG is called to reset the cell flags. The time is updated and the new time step calculated. The tests are made to determine if an edit should be made, if so, subroutine EDIT is called; and if the problem should be stopped subroutine SAM is called to write COMMON on tape for restarting the problem in the future and subroutine OFFMON is called to exit the problem from the machine and do the systems bookkeeping to clean out files. If the problem is to continue then calculations of the new values of ETA and PSI proceed to start the next computational cycle.

Subroutine GEN is called only at the start of a problem to read the input data and do the necessary calculations to load the arrays with the initial conditions of the problem. Subroutine GENS is called to load the

arbitrarily shaped Lagrangian boundaries. Subroutine REFLAG is called to set the cell flags for the start of the calculations.

Subroutine GENS calculates the coordinates of the surface particles. This subroutine is only called if the value of ISURFACEOPT is 1 on data card 3.

Subroutine GENB calculates the position of the arbitrarily shaped Lagrangian boundaries. This subroutine generates the coordinates of the center of each Lagrangian boundary segment in each cell and the direction numbers of the normal vector at the center of each boundary segment in each cell. The normal vector points into the fluid.

Subroutine PRES solves the pressure iteration Eq. (9) or Eq. (14) for the case where the cell contains a Lagrangian boundary. Subroutine VEL is called to calculate the advanced time velocity in each cell from the pressure gradient for all cells in the entire mesh. During the course of calculating new pressures at each cell during the iteration, subroutine VELP is called to calculate new advanced time velocities in individual cells as the pressure calculation progresses through the mesh. The pressure iteration proceeds until the ratio of the square of the norm of the matrix of pressure residuals to the square of the norm of the pressure matrix is less than the value of EPS2. The value of EPS2 is specified in optional data card 3. If the ratio of norms cannot be made less than EPS2 within 200 iterations, then the problem is stopped by calling EDIT and OFFMON.

Subroutine TRAJ moves the marker particles and does the bookkeeping to keep track of the coordinates of each particle and the number of particles in each cell. If surface particles are used in addition to the marker particles

this subroutine also moves them and keeps track of their coordinates.

Subroutine FLOWIN introduces new marker particles at the inflow boundaries as the old particles move into the computational mesh.

Subroutine SHIFT moves the arbitrarily shaped Lagrangian boundaries. Subroutine MOVE is called to calculate the velocities of the boundary points. Based on these velocities, new boundary coordinates are calculated in SHIFT, and from here on subroutine SHIFT essentially a suplicate of GENB. The coordinates of the center of each Lagrangian boundary segment and the direction numbers of the normal vector are calculated for each cell. The Lagrangian boundary cell flags are reset. Subroutine SHIFT only calls MOVE and calculates new position and normals for the Lagrangian boundaries if parameter MOPT specified in data card 3 is equal to 1, if MOPT is 0 the calculations are omitted.

Subroutine MOVE calculates the new velocities for the boundary points specified in data card 15. Subroutine MOVE also sets MOPT = 0 after the boundary motion ceases in order to avoid unnecessary calculations in subroutine SHIFT.

Subroutine REFLAG resets the cell flags based on the new coordinates of the marker particles. If no surface particles are used the pressure in the cells flagged as surface cell is set to 0. If surface particles are included the pressure in the surface cells is interpolated by straight line interpolation between the pressure in the adjacent full cell and zero pressure at the actual position of the free surface as specified by the locus of surface particles.

Subroutine VEL calculates the horizontal and vertical components of velocity in every cell from the pressure gradient as in Eq. (5) and (7).

VEL 55 is called to calculate the velocities in the free surface cells.

Subroutine VELP is the same as subroutine VEL except that it calculates the velocity only for a single specific cell.

Subroutine VEL 55 calculates the components of velocity in all the free surface cells such that the divergence of the velocity vanishes.

Subroutine VEL55P is the same as subroutine VEL55 except the calculation is done only for a single specific cell.

Subroutine VEL57 calculates the velocity in cells containing the intersection of a free surface and Lagrangian boundary. VEL56 is called to calculate local velocity at specific points.

Subroutine VEL56 calculates local velocity at a specific point and uses the same weighting formula as used in TRAJ, VEL and VELP.

Subroutine EDIT causes the data for each cell to be listed. A detailed description of the format of the edit is given in appendix C. Subroutine EDIT also causes the cathode ray tube (CRT) pictures to be made.

Subroutine SAM saves COMMON on tape so it can be used for restarting the problem.

Subroutine RXD reads a tape containing COMMON for restarting a problem.

Subroutine OFFMON causes the problem to exit from the machine.

FIG. /

BIBLIOGRAPHY

- Amsden, Anthony A., and Francis H. Harlow, The SMAC Method: A Numerical Technique for Calculating Incompressible Fluid Flows, Los Alamos Scientific Laboratory, Rept. LA-4370, 1970.
- Chan, Robert K.-C., and Robert L. Street, "A Computer Study of Finite-Amplitude Water Waves", *J. Computational Phys.*, 6, (1), 68-94, 1970.
- Chan, Robert K.-C., Robert L. Street, and Theodor Strelkoff, Computer Studies of Finite-Amplitude Water Waves, Dept. of Civil Engineering, Stanford Univ., Stanford, Calif., Tech. Rept. 104, 1969.
- Chorin, Alexandre Joel, "Numerical Solution of the Navier-Stokes Equations", *Math. Computations* 22, 745-762, 1968.
- Fromm, Jacob, E., A Method for Computing Nonsteady, Incompressible, Viscous Fluid Flows, Los Alamos Scientific Laboratory Rept. LA-2910, 1963.
- Garcia, W. J., A Study of Water Waves Generated by Tectonic Displacements, College of Engineering, Univ. of California, Berkeley, Hydraulic Engineering Laboratory Rept. HEL 16-9 (Ph.D. Thesis), 1972.
- Harlow, Francis H., and John P. Shannon, "The Splash of a Liquid Drop", *J. Appl. Phys.*, 38, (10), 3855-3866, 1967.
- Harlow, Francis H., and J. Eddie Welch, "Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface", *Phys. Fluids*, 8, (12), 2182-2189, 1965.
- Harlow, Francis H., and J. Eddie Welch, "Numerical Study of Large-Amplitude Free-Surface Motions", *Phys. Fluids*, 9, (5), 842-851, 1966.
- Hirt, C. W., "Heuristic Stability Theory for Finite-Difference Equations", *J. Computational Phys.*, 2, 339-355, 1968.

- Hirt, C. W., and Francis H. Harlow, "A General Corrective Procedure for the Numerical Solution of Initial-Value Problems", J. Computational Phys. 2, 114-119, 1967.
- Hirt, C. W., and J. P. Shannon, "Free-Surface Stress Conditions for Incompressible Flow Calculations", J. Computational Phys., 2, 403-411, 1968.
- Nichols, B. D., Recent Extensions to the Marker-And-Cell Method for Incompressible Fluid Flows, Los Alamos Scientific Laboratory, Rept. LA-DC-11631, 1970.
- Viecelli, James A., "A Method for Including Arbitrary External Boundaries in the MAC Incompressible Fluid Computing Technique", J. Computational Phys. 4, 543-551, 1969.
- Viecelli, James A., "A Computing Method for Incompressible Flows Bounded by Moving Walls", J. Computational Phys. 8, 119-143; 1971.
- von Neumann, J. and R. D. Richtmyer, "A Method for the Numerical Calculation of Hydrodynamic Shocks", J. Appl. Phys. 21, 232-237, 1950.
- Welch, J. Eddie, Francis H. Harlow, John P. Shannon, and Bart J. Daly, The MAC Method: A Computing Technique for Solving Viscous, Incompressible, Transient Fluid-Flow Problems Involving Free Surfaces, Los Alamos Scientific Laboratory, Rept. LA-3425, 1966.

APPENDIX A: DEFINITION OF SYMBOLS

The following is a list of the variables contained in COMMON. COMMON is contained in subroutine MAIN.

KYA	maximum number of computational mesh cells.
KQED	maximum number of computational mesh cells in any row or column.
KPAT	maximum number of marker particles.
KNBR	maximum number of intersection points between Lagrangian arbitrary boundaries and computational mesh grid lines.
NSP	maximum number of surface particles.
COMMON 1	
NVAC	not used in this version
NGEN	flag for specifying centered or backward differencing for the advection terms of the Navier-Stokes equation, value specified on data card 23.
UAC	not used in this version.
VAC	not used in this version.
TM1	not used in this version.
TM2	not used in this version.
TM3	not used in this version.
COMMON 2	
NCYCLE	computational cycle number.
NSTOP	cycle number at which calculations are to stop value, specified on data card 24.
NEDTM	cycle number of first edit after cycle 0, value specified on data card 24.

NEDIT	number of cycles between edits; value specified on data card 24.
OPT	a flag to specify inclusion on cell flag and boundary data in edit value 1. or 0. specified in data card 22.
NXQQ	the upper end of inclusive limits on the sequential numbers of boundary points which are to be moved in subroutine SHIF1', value specified on data card 15.
MOVIE	not used in this version.
COMMON 3	
TIME	problem time, calculated in MACMAIN.
EDTIM	time of first edit after 0, value specified on data card 22.
EDDT	time between edits, value specified on data card 22.
NSPART	number of surface particles in problem.
ZSURF	vertical coordinate of free surface used to initialize hydrostatic pressure in mesh, value specified on data card 20.
ISURFACEOPT	flag to specify inclusion of surface particles, value 1 or 0 specified on data card 3.
KSTOP	sentinal flag to stop reading surface particle sequence, value specified on data card 13.
NSEQ	sentinal flag to stop reading Lagrangian boundary point sequence, value specified on data card 14.
COMMON 4	
EPS2	maximum value of ratio of squares of norm.of residuals of pressure to norm of pressure value specified on data card 23, Transferred to PRES as EPS.

BETA	value of relaxation parameter for overrelaxation of pressure iteration equation its maximum value if 0.5 but should be less, value specified on data card 23.
DT	time step between computational cycles, calculated in MACMAIN.
VMAX	maximum value of sum of absolute values horizontal and vertical cell velocity components over the mesh calculated in MACMAIN.
UMAX	sum of absolute values of horizontal and vertical cell velocity components in any cell calculated in MACMAIN.
RELAX	the relaxation coefficient for pressure iteration equation, calculated in MACMAIN.
CUT	
DTR	initial and maximum time step, specified on data card 22.
COMMON 5	
SMIN	minimum fraction of cell on fluid side of Lagrangian arbitrary boundary to allow cell to be treated individually, if less fluid than SMIN cell is blended with adjacent cell, SMIN should be about 10% to 25% of cell area value specified on data card 4 for use in GEN and again on data card 23.
EPS1	boundary sensing parameter equal to a fraction of cell width, cause flag ARB to be set if marker particles are within EPS1 of Lagrangian boundary and fluid is treated as if it is in contact with boundary, value specified on data card 4 for use in GEN and again on data card 23.

NBMAX

MOPT flag to specify moving boundary option value 1 or
0 specified on data card 3.

UBTO not used in this version

VBTO not used in this version

COMMON 6

UM equal to product of viscosity and time step, calculated in MACMAIN.

MU kinematic viscosity, value specified on data card 5.

RHOA density of fluid, value specified on data card 5.

GR body acceleration in horizontal direction positive
to right, value specified on data card 5.

GZ body acceleration in vertical direction (gravity),
positive downward, value specified on data card 5.

G magnitude of vector sum of horizontal and vertical
body acceleration, calculated in GEN.

ZHRT not used in this version

PI $\pi = 3.14159265358979$

ISEC total number of intersections of Lagrangian boundary
section with Eulerian mesh.

COMMON 7

VIN vertical component of specified inflow boundary velocity,
value equal to VVEL read on data card 11.

UIN horizontal component of specified inflow boundary
velocity, value equal to UVEL read on data card 11.

DKP distance between marker particles in horizontal
direction.

DLP distance between marker particles in vertical direction.

NBP	total number of input particles lying just outside an inflow boundary, value specified on data card 12.
DDR	horizontal spacing of input particles lying just outside an inflow boundary, value specified on data card 12.
DDZ	vertical spacing of input particles lying just outside an inflow boundary, value specified on data card 12.
COMMON 8	
NEXP	value of exponent specifying slab or cylindrical geometry, 0 for slab, 1 for cylindrical, value specified on data card 3.
NEXPP	equal to NEXP+1.
NPART	total number of marker particles.
NDIV	equal to NPART.
XS(10)	array of horizontal coordinates of end points of lines of surface particles, values specified on data card group 13. Maximum size of array is 10.
YS(10)	array of vertical coordinates of end points of lines of surface particles, values specified on data card group 13. Maximum size of array is 10.
DSP(10)	array of linear distance between surface particles values read in data card group 13.
COMMON 9	
DEBUGA DEBUGB DEBUGC DEBUGC	variables specified for debugging, not used in this version.
XW(35)	array of displacement coordinates for moving boundary, value specified on data card 19. Maximum size of the array is 35 values.

UW(35)	array of velocity values for moving boundary calculated in GEN.
NW	number of boundary displacement-time coordinates.
IT	counter for boundary displacement-time coordinates.
COMMON 10	
NGRID	total number of computational mesh cells.
KMAX	total number of computational mesh cells in horizontal direction.
LMAX	total number of computational mesh cells in vertical direction.
KMAXZ	equal to KMAX-1.
LMAXZ	equal to LMAX-1
KMAXZZ	equal to KAMX-2
LMAXZZ	equal to LMAX-2
COMMON 11	
KDMN	cell number to start edit in horizontal direction, value specified on data card 17.
KDMX	cell number to end edit in horizontal direction, value specified on data card 17.
LDMN	cell number to start edit in vertical direction, value specified on data card 17.
LDMX	cell number to end edit in vertical direction, value specified on data card 17.
MESS(7)	identification line for edit, specified on data card 2.
ACF(10)	not used in this version
NXQM	

LXQM	lower end of inclusive limits on the sequential numbers of boundary points which are to be moved in subroutine SHIFT, value specified on data card 15.
AP	not used in this version
BP	not used in this version
COMMON 12	(CRT plot variables values specified on data card 16)
ICRT	variable to indicate if grid should be plotted on CRT pictures.
XMAX	relative picture size in horizontal direction maximum value = 0.9.
YMAX	relative picture size in vertical direction maximum value = 0.9.
CRTRA	horizontal coordinate to start plot.
CRTRB	horizontal coordinate to stop plot.
CRTZA	vertical coordinates to start plot.
CRTZB	vertical coordinate to stop plot.
COMMON 13	
ITEST	not used
ICZ	not used
IZ	working index
GAMA	ratio of specific heats for gas in a bubble, specified on data card 6.
PGAS	not used
PGASZ	gas pressure in bubble, specified on data card 6.
PAMB	ambient pressure, specified on data card 6.
GASV	not used
GASVZ	not used

COMMON 14

NF	not used
NG	not used
NIT	not used
H	not used
XG	not used
HH	not used
GG	equal to vector sum of body accelerations (=G)
FUDGE	not used
NTAL	not used
CRAX	parameter defined in MACMAIN.
DELZ	not used

COMMON 15

BPX(KNBR)	array of horizontal coordinates of boundary of region with marker particles values specified in data card group 10.
PTX(KNBR)	array of horizontal coordinates of intersections of Lagrangian boundaries and computational grid, calculated in GENB and SHIFT.
RMP(KNBR)	array of horizontal coordinates of boundary intersections similar to PTX.
RNP(KNBR)	array of horizontal coordinates of boundary intersections similar to PTX.
XB(20)	array of horizontal coordinates of end points of Lagrangian boundary segments, values specified in data card group 14. Maximum size of array is 20 values.

COMMON 16

BPY(KNBR) array of vertical coordinates of boundary of region with marker particles, values specified in data group 10.

PTY(KNBR) array of vertical coordinates of intersections of Lagrangian boundaries and computational grid, calculated in GENB and SHIFT.

ZMP(KNBR) array of vertical coordinates of boundary intersections similar to PTY.

ZNP(KNBR) array of vertical coordinates of boundary intersections similar to PTY.

YB(20) array of vertical coordinates of end points of Lagrangian boundary segments, values specified in data card group 14.

COMMON 17

DS(KNBR) array of distances between Lagrangian boundary segment end points XB, YB and intersections points RMP, ZMP, calculated in GENB and SHIFT.

IBP(KNBR) a structured array containing NBC and IPX.

VLG(KNBR) array of vertical components of Lagrangian boundary velocity, calculated in MOVE.

ULG(KNBR) array of horizontal components of Lagrangian boundary velocity calculated in MOVE.

VTMP(KNBR) array of vertical components of an intermediate step velocity used in calculation of new moving boundary position, in SHIFT.

COMMON 18

R(KQED) array of horizontal coordinates of center of computational cell calculated in GEN.

Z(KQED) array of vertical coordinates of center of computational cells, calculated in GEN.

RB(KQED) array of horizontal coordinates of right boundaries of computational cells, calculated in GEN.

ZB(KQED) array of vertical coordinates of top boundaries of computational cells, calculated in GEN.

DR(KQED) array of horizontal dimensions of cells, values specified in data card group 7.

DZ(KQED) array of vertical dimensions of cells, values specified in data card group 8.

COMMON 19

DRB(KQED) array of center to center distance of cells in horizontal direction, calculated in GEN.

DZB(KQED) array of center to center distance of cells in vertical direction, calculated in GEN.

UTMP(KNBR) array of horizontal components of an intermediate step velocity used in calculation of new moving boundary position, in SHIFT

PVX(KNBR) array of horizontal components of an intermediate velocity, similar to VTMP, in SHIFT.

PVY(KNBR) array of vertical components of an intermediate velocity, similar to VTMP, in SHIFT.

COMMON 20

FLAG(KYA) a structured array of all flags for each cell, the structure is specified in MAIN. Contents of FLAG as follows. Bits 1 or 0 are 1 on or yes, 0 off or no.

INTRF	not used
CNTRB	not used
SUR	1 or 0, cell is a free surface cell, has particles and adjacent to empty cell.
FULL	1 or 0, cell is full, has particles and not adjacent to empty cell.
EMP	1 or 0, cell is empty.
BND	1 or 0, cell is a boundary of mesh cell, top, bottom, right and left row of cells.
IN	1 or 0, cell is an inflow boundary cell.
OUT	1 or 0, cell is an outflow boundary cell.
FRSLP	This flag is not used in this version.
NOSLP	This flag is not used in this version
EMPEND	This flag is not used in this version
OB	1 or 0, cell contains a Lagrangian boundary.
COR	1 or 0, cell adjacent to or containing Lagrangian boundary but is empty and is not treated separately in calculations.
OK	1 or 0, cell has Lagrangian boundary and can use relaxation boundary condition to calculate pressure.
NPRTA	the number of particles in the cell, listed as NPT in edit.
NPRTB	a pointer specifying number of cell to look to for data for calculations of certain COR cells, listed as KMPR in edit.
MPRTA	the number of particles in the cell, as calculated in TRAJ.
MPRTB	not used
GAS	1 or 0, used only in bubble calculations.

ARB 1 or 0, particles are within EPSI of Lagrangian
 boundary and fluid is assumed to be in contact
 with boundary for pressure calculation.

SNC 1 or 0, cell is SUR cell and not COR, flag is used
 in VEL57 only.

COMMON 21

RP(KPAT) array of coordinates of marker particles this is a
 packed word, packed in CLICHE PACK.

SPX(NSP) array of horizontal coordinates of special surface
 particles.

SPY(NSP) array of vertical coordinates of special surface
 particles.

COMMON 22

UAVG(KYA) array of horizontal and vertical components of average
 particle velocity in each cell, this is a packed
 word, packed in CLICHE PACK.

UBND(KYA) array of horizontal components of velocity of Lagrangian
 boundary segment in each cell.

VBND(KYA) array of vertical components of velocity of Lagrangian
 boundary segment in each cell.

COMMON 23

P(KYA) array of cell pressure for each cell, pressure at
 center of all cells, except those containing
 Lagrangian boundary (OK) then pressure at center
 of boundary segment.

COMMON 24

U(KYA) array of horizontal components of velocity in each
 cell, specified at right center of cell.

ETA(KYA) array of horizontal components of sum of old velocity,
 advection terms, viscous times and body force terms
 for each cell, calculated in MACMAIN.

COMMON 25

V(KYA) array of vertical components of velocity in each cell, specified at top center of cell.

PSI(KYA) array of vertical components of sum of old velocity, advection terms, viscous terms and body force terms for each cell, calculated in MACMAIN.

COMMON 26

DNOR(KYA) array of horizontal component of direction number of normal from center of Lagrangian boundary segment in each cell, pointing into fluid, calculated in GENB and SHIFT.

DNOR(KYA) array of vertical component of direction number of normal from center of Lagrangian boundary segment in each cell, pointing into fluid, calculated in GENB and SHIFT.

PNORX(KYA) array of horizontal coordinates of center of Lagrangian boundary segment in each cell calculated in GENB and SHIFT.

PNORY(KYA) array of vertical coordinates of center of Lagrangian boundary segment in each cell calculated in GENB and SHIFT.

COMMON MARY

IB tape unit number for I/O routines

NHC, NHD, NHE dump tape names specified on data card 21.

NTAPE name of tape to be written at end or read for restart.

NPT1, NPT2, flags to indicate which of three tapes available is
NPT3, NIPE to be read or written

COMMON GOBCOM

IGOB I/O identification information for system

APPENDIX B: INPUT CARD FORMAT FOR DATA

The following is a listing of the data input for the ABMAC Eulerian incompressible hydrodynamic computer code at LLL, Livermore. This list of data specifically applies to the version of ABMAC called MAB31 but is also applicable to other versions. The data is read computer code from a file named DATA.

Card or
Card Group
No.

Card Format

Parameter Names

1.

110, 3A10

NIXE, ID1, ID2, ID3

NIXE is 1 or 2 indicating new problem or restart respectively and ID1, ID2, ID3 are identification parameters for the LLL system, ID2 is problem name, ID1 and ID3 are output box numbers. This card is read in MACMAIN.

2.

7A10

MESS

This is problem identification and is printed out at the top of each section of the edit. Card read in GEN.

3.

7I10

KMAX, LMAX, NEXP, MOPT, ISURFACEOPT

KMAX and LMAX are the number of computational cells in the horizontal and vertical directions respectively. NEXP is an exponent to specify plane or cylindrical symmetry, 0 for plane and 1 for cylindrical. MOPT is a flag for the moving boundary option, 0 boundary does not move, 1 indicates boundary moves and code goes through subroutine SHIFT each cycle. ISURFACEOPT is a flag for inclusion of surface particles 0 indicates

surface particles not included and code skips all calculations involving the surface particles, 1 indicates surface particles are included. Card read in GEN.

4. 7E10.3 SMIN, EPS1

SMIN is sensing parameter equal to a fraction of the area of a computational cell (from 1/10 to 1/4) such that in the arbitrary boundary option if there is less fluid in the cell than SMIN, the cell is blended with an adjacent cell for computations. EPS1 is the boundary sensing parameter equal to a fraction of a cell width, this causes flag ARB to be set if the marker particles are within distance EPS1 of the arbitrary boundary and the fluid is treated as if it is in contact with the boundary, otherwise the fluid is assumed to be separated from the boundary and the pressure in the cell is set to zero. Card read in GEN.

5. 7E10.3 GZ, GR, MU, RHOA, RHOB

The vertical and horizontal acceleration (gravity) fields are GZ and GR respectively, for GZ positive is downward so gravity is entered as a positive acceleration, for GR positive is to the right. MU is the kinematic viscosity, RHOA and RHOB are fluid densities, for one fluid both are the same, RHOB would be the second fluid density for use in a two fluid version of the code. Card read in GEN.

6. 7E10.3 PAMB, PGASZ, GAMA

These parameters are for use in gas bubble problems where PAMB is the ambient free surface pressure, PGASZ is the

bubble pressure and GAMA is the ratio of specific heats for the bubble gas. In problems with no bubbles enter zeroes.

Card read in GEN.

7. 2I10, E10.3 M, N, XX

This card enters computational cell size in horizontal direction. Cell width XX is entered from K = M through K = N. If all cells have some horizontal dimension, then this card is followed by a blank card or M = 0 otherwise the next region of cell size in horizontal directions would be read. Cards read in GEN.

8. 2I10, E10.3 M, N, XX

This card enters computational cell size in vertical direction. Cell height XX is entered from L = M through L = N. If all cells have same horizontal dimension, then this card is followed by a blank card or M = 0 otherwise the next region of cell size in vertical direction would be read. Cards read in GEN.

9. 7I10 NP, KA, KB, LA, LB

This card caused to be loaded marker particles with linear density NP (total particles in cell is NP x NP) from K = KA through K = KB in horizontal direction and from L = LA through L = LB in vertical direction. If this card is blank or NP = 0 then routine skips to card 11 otherwise continues to the next card (10.). Card read in GEN.

10. 2E10.3 BPX(M), BPY(M)

This is a sequence of M points defining the boundary of the fluid (particle) region. Convention is that the region lies to the left as one moves from BPX(1), BPY(1) to BPX(2), BPY(2), etc. GEN continues reading card group 10 until the initial values BPX(1) and BPY(1) are again encountered, then it goes back to card 9. Cards read in GEN.

11.	4I10, 2E10.3, 2I5	KA, KB, LA, LB, UVEL, VVEL, KIN, KOUT
-----	-------------------	--

Loads inflow and outflow boundary velocities with this data. Horizontal velocity UVEL and vertical velocity VVEL are load in boundary cells from $K = KA$ through $K = KB$ and from $L = LA$ through $L = LB$. The inflow boundary flag IN is set according to KIN, 0 no inflow boundary or 1 inflow boundary the outflow boundary flag OUT is set according to KOUT, 0 no outflow boundary or 1 outflow boundary. Continue reading additional sequences until a blank card or $KA = 0$ is found then goes on to next card. Cards read in GEN.

12. I10, 4E10.3 NBP, XXA, YYA, DDR, DDZ

This card specifies a line of input particles for inflow boundaries lying just outside the inflow boundary where NBP is the total number of particles XXA, YYA are the coordinates of the first particle and DDR and DDZ are the horizontal and vertical particle spacing respectively. Continue reading additional sequences until a blank card or NBP = 0 is found, then goes on to next card. Cards read in GEN.

13. 3E10.3, I10 XS(M), YS(M), DSP(M), KSTOP

This card specifies coordinates of surface particles, XS(1), YS(1) are the coordinates of the first surface particle, the particles are loaded in a straight line with spacing DSP(1) to the next coordinate XS(2), YS(2), etc. Continues reading coordinates in this group of cards until it finds KSTOP is non zero then it goes on to the next group of cards. Cards read in GENS. If the flag ISURFACEOPT = 0 then this sequence is skipped and this group of cards should be omitted.

14. 2E10.3, I10 XB(N), YB(N), NSEQ

This card specifies the coordinates of end points of boundary segments forming a continuous section of boundary. XB(N) and YB(N) are the horizontal and vertical coordinates for the boundary segments. NSEQ is a flag, if 0 then the next set of coordinates is read, a 1 indicates that the boundary sequence is complete and there are no more points to be read, a 2 indicates that this section of boundary is complete and is to be processed but there is another section of boundary yet to be read and processed separately. This option allows numerous boundary sections to read in without the sections being continuous. The points XB(N), YB(N) are to be ordered such that as one proceeds along the boundary, the fluid is to the left. Cards are read in GENB.

15. 6I10 LXQM, NXQQ

These are the inclusive numbers of the boundary points which are to move in the moving boundary option. Card read in GEN.

16. I10, 6E10.3 ICRT, CRTA, CRTB, CRTZA, CRTZB,
XMAX, YMAX

This card gives data for plotting the cathode ray tube (CRT) pictures for data display for use in the LLL system. ICRT indicates if a grid is to be plotted, if 1 then no grid, if 5 then grid is plotted. CRTRA and CRTRB are the start and stop coordinates of the plot in the horizontal direction; CRTZA and CRTZB are the start and stop coordinates of the plot in the vertical direction. XMAX and YMAX tell the size of the picture they have a maximum value of 0.9 for full page, XMAX gives the horizontal and YMAX the vertical dimension.

Card read in GEN.

17. 6I10 KDMN, KDMX, LDMN, LDMX

These are the limits of the cell numbers for the edit of the data for each cell. KDMN is the start and KDMX is the stop cell number in the horizontal direction. LDMN is the start and LDMX is the stop cell number in the vertical direction.

Card read in GEN.

18. I10 NW

NW is the number of points in the time-displacement data to follow. Card read in GEN.

19. 2E10.3 TW(M), XW(M)

TW(M) is the time and XW(M) is the horizontal displacement coordinate for calculating the horizontal velocity of the moving boundary. These cards are read in GEN and the velocity is calculated between sequential points. The time and velocity data are used to calculate the new coordinates of the boundary as it moves.

20. 2E10.3 ZSURF

This is the vertical coordinate of the free surface for calculating the hydrostatic pressure field at the start of the calculation. Card read in GEN.

21. 3R5 NHC, NHD, NHE

These are the names of three tapes for storing the data from common for restarting the problem. Three tapes provide a backup if one or two are bad. If only one tape is used the tape name NHC is repeated 3 times. If no restart tape is used then a blank card is inserted here. Card read in MACMAIN.

22. I10, 6E10.4 1, DTR, EDDT, EDTIM, OPE

Optional data card 1, DTR is initial and maximum time step for cycles. EDDT is time between edits, EDTIM is time of first edit after 0. OPE is flag to include flag and moving boundary data in edit, 0. to exclude, 1. to include. Card read in MACMAIN.

23. I10, 6E10.4 2, EPS1, EPS2, BETA, NGEN, SMIN

Optional data card 2, EPS1 is the boundary sensing parameter which is also entered on card 4. EPS2 is a parameter indicating size of the pressure error upon final relaxation of the pressure equation in subroutine PRES, this parameter is normally about 0.0008. BETA is a relaxation parameter for the over-relaxation of the pressure equation, its maximum value is 0.5 for stability, it should be less than or equal to about 0.46. NGEN is a flag 0. or 1. to indicate the differencing scheme for the advection terms in the Navier-Stokes equation, 0. indicates backward differencing and 1. indicates centered differencing. SMIN is the sensing parameter for cell area, the same as entered on card 4. Card read in MACMAIN.

24. I10, 6E10.4 3, NSTOP, NLIMIT, NEDIT, NEDTM

Optional data card 3. NSTOP is the cycle number of the last cycle to be run. NLIMIT is not used at set equal 0. NEDIT is the number of cycles between edits if cycle number is used to determine edit times. NEDTM is the number of the cycle for the first edit after the 0 cycle. Card read in MACMAIN.

25. I10, 6E10.4 4, DEBUGA, DEBUGB, DEBUGC, DEBUGD

Optional data card 4. Debug options, not used. Card read in MACMAIN.

26. I10, 6E10.4 5, KDMN, DKMX, LDMN, LDMX

Optional data card 5. Limits for the range of cells to be printed out in edit used to change data of card 17. Card read in MACMAIN.

27. I10, 6E10.4 6, CRTRA, CRTRB, CRTZA, CRTZB, XMAX, YMAX

Optional data card 6. Determines limit and size of CRT pictures used to change data of card 16. Parameter ICRT for grid lines cannot be changed. Card read in MACMAIN.

The first three optional data cards are necessary to start the problem since all contain data necessary to the problem and not given anywhere else. Optional data card 4 is omitted. Optional data cards 5 and 6 are omitted at start of problem but may be used at restart to change edit or CRT pictures. The input data deck should be ended with 2 blank cards.

The problem may be stopped with SW1. at any time at which time the tape specified in card 21 is written with the data necessary to restart the problem at that point. In order to restart, the first restart card is card 1 with the digit 2 in the first field I10. The second card is the tape name card, card 21 which now specifies the tape to be read for restart, the first tape name is read to restart. If optional data cards are to be used to make a change, then these follow the tape name card. The deck is ended with 2 blank cards. On restart only data read in subroutine MACMAIN is entered.

APPENDIX C: OUTPUT EDIT FORMAT

The following are instructions for reading the output of the ABMAC computer code as used at Lawrence Livermore Laboratory.

The first page of printout before the beginning of the edit is the listing of the data as read into the code to generate the problem. Two examples are shown, example 1, a problem without surface particles, Fig. 1, and example 2, a problem with surface particles, Fig. 2. The surface particles serve to increase the surface resolution for problems with small surface motion.

Referring to Fig. 1, Example 1 sample edit of input data for problem without surface particles. Line 1 is identification for the I/O routines at LLL. Line 2 is identification for the problem and is also printed out at the start of each section of edit. From line 3 on, specific reference is made to example 1. Line 3 tells that the horizontal extent of the computational grid is 150 zones and the vertical extent is 40 zones; the next 0 is a flag for the geometry of problem, 0 for plane geometry as in this case or 1 for cylindrical geometry; the next 1 is a flag for the moving boundary option, 0 for no moving boundary or 1 for a moving boundary as in this case; the next 0 is a flag for the inclusion of surface particles, 0 for no surface particles as in this case or 1 for surface particles to be included. Line 4 contains the sensing parameters for merging a boundary cell and for putting the fluid in contact with the boundary; in this case 3.6×10^{-4} is 1/10 cell area and if in this problem a boundary cell contains less than 1/10 cell of fluid that cell is merged with an adjacent cell for the calculation; the 6.0×10^{-2} is

the cell width in this case, but in general is the maximum distance from the boundary in which marker particles are assumed to be in contact with the boundary for purposes of pressure calculation. Line 5 gives the acceleration field, viscosity and density; English units are being used in this problem; 3.22×10^1 is the acceleration of gravity in ft/sec^2 , positive is down for the vertical acceleration field; next is the horizontal acceleration, 0 in this case; next the kinematic viscosity of the fluid, $10^{-5} \text{ ft}^2/\text{sec}$; next the density of the fluid, $1.94 \text{ slugs}/\text{ft}^3$, the number is repeated since there is also a provision for 2 fluids in one version of this code and the density of the second fluid would be put in the second place. Line 6 is the ambient pressure, 0. here; the pressure of a gas bubble, 0; and gamma for the gas in the bubble. Line 7 gives the horizontal zone sizes; inclusive zones 1 through 150, each zone is $6 \times 10^{-2} \text{ ft}$ wide. Line 8, the zeros tell the generator to read next the vertical zones. Line 9, the vertical zone sizes; inclusive zones 1 through 40, each zone is $6 \times 10^{-2} \text{ ft}$. high. Line 10, the zeros indicate to go on to next data. Line 11 gives particle density for given zones; in this case particles have linear density of 3 particles per zone for zones 3 through 25 horizontally and 2 through 10 vertically; area particle density for zone is $3 \times 3 = 9$ particles per zone. Lines 12 through 16 are the coordinates of the area which is actually occupied by the fluid having the particle density given on line 11; coordinates are x and y moving along outside of fluid such that the fluid is always on the left; the last coordinates are a duplication of the first. Line 17 gives linear particle density of 2 per zone (area density is $2 \times 2 = 4$) for zones 26 through 149 horizontally and 2 through 10 vertically. Lines 18 through 22 are coordinates

of region of fluid as lines 12 through 16. Line 23 is zero telling to go on to next data. Line 24 gives inflow and outflow boundary velocities; here zones 1 through 150 horizontally and 1 through 40 vertically have horizontal velocity component 0. and vertical velocity component 0.; the end two zeros are flags first one 0 for no inflow boundary and 1 for inflow boundary and 1 for inflow boundary and second 0 for no outflow boundary and 1 for outflow boundary. Line 25 is zero telling to go on to next data. Line 26 gives particle density and coordinates of particles lying outside inflow boundaries, 0 in this case, no inflow boundaries. Lines 27 and 28 are coordinates of end points of boundary line segments given so that fluid is on left; the 1 at end of line 28 is a flag indicating it is last coordinate to be read, a 0 means read more and a 2 means to process this boundary and then go back and read another series of boundary segments. Line 29 gives the inclusive numbers of boundary points to be moved. Line 30 gives I/O information for the cathode ray tube (CRT) pictures which are to be made; 5 means to include a grid; 0. and 8.88 are horizontal coordinates of left and right edge of picture; 0. and 2.28 are vertical coordinates of bottom and top of picture; 9×10^{-1} and 3×10^{-1} are parameters for the CRT system for proportioning the size of the picture, 3 times as wide as it is high. Line 31 tells zones to be edited in printout edit; here it is entire grid, zones 1 through 150 horizontally and 1 through 40 vertically. Lines 32 through 34 give boundary motion data N is point number; TW(N) is time; XW(N) is horizontal position; UW(N) is velocity calculated in generator, only velocity is actually used to move boundary during calculations, time and coordinates are not used beyond

calculating boundary velocity in generator. Line 35 is vertical coordinate of water surface for calculation of hydrostatic pressure in generator, 5×10^{-1} ft here. Line 36, blank here would have the name of the tape for stopping and restarting the problem. Line 37 is optional data card 1; gives initial time step 6×10^{-3} sec; time between edits, 1×10^{-1} sec.; time of first edit after zero time, 1×10^{-1} sec; and a flag for printing out only velocity and pressure data in edit, a 0. or a 1. as here to print out velocity and pressure data and all flags and boundary data for all cells. Line 38 is optional data card 2; gives boundary sensing distance as in line 4, 6×10^{-1} ft; parameter to indicate how precisely the pressure iteration is to be carried out, 8×10^{-4} here, this is the ratio of the norm of the residuals in the pressure matrix to the norm of the pressure matrix; the over relaxation parameter, which determines numerical stability is 4.1×10^{-1} here; next is a flag for the differencing scheme to be used for the advection terms of the Navier-Stokes equations, 0. for backward differencing and 1. for centered differencing, backward differencing is being used for this problem; and the minimum cell area parameter as in line 4, 3.6×10^{-4} in this case or 1/10 zone area. Line 39 is optional data card 3; tells the maximum number of cycles to run, 500. here; an unused 0.; the cycles between edits; and the cycle after 0 for first edit if problem is to be edited by cycle rather than by time as is the case for this problem. This ends the input data for the problem. The next part of the edit starts the cycle edit, and the first line of the cycle edit is cycle number, time of cycle and time step for that cycle. The surface particle coordinates are printed next. If no surface

particles are used as for example 1 then these coordinates are the coordinates of all marker particles located in cells flagged surface cells, and as a result many coordinates are not actually at the surface and are for extraneous particles.

The data of example 2 shown in Fig. 2 is the same format as example 1 with the exception of the addition of surface particles. Example 2 uses metric units, the computational grid is 252×18 zones and each zone is 300 m square. The fluid is broken into 5 regions containing 1 or 2 particles linear density, (1 or 4 interior marker particles per zone); the surface particles are not included here. The surface particle data start at line 49 of example 2, just below the inflow-outflow boundary data; the surface particle coordinates are calculated from end points and particle spacing as given on lines 49 and 50 of example 2. Here the surface particles start at $x = 7.495 \times 10^4$ m and $y = 4.3 \times 10^3$ m and are 7.5×10^1 m apart ending at $x = 2.5 \times 10^1$ m and $y = 4.3 \times 10^3$ m; the 1 indicates end of surface particle sequence. Following surface particle data are boundary coordinates as in example 1. There is an extra line added before the boundary motion time-displacement data, that is the total number of interior marker particles and surface particles, 6953 interior and 1000 surface particles here. The other difference from example 1 is the surface particle coordinates, here in example 2 the coordinates are actually the locus of points along the fluid surface, the coordinates of marker particles contained in surface cells are not printed out if surface particles are used.

Following the listing of surface particle coordinates is printed the problem identification line, this is shown at the top of Fig. 3. Continuing

reference to Fig. 3, the next line gives the vertical coordinate of the center of the row of cells for which the data is presented ($Z =$), the cycle number and the time of the cycle. The next line gives the row number ($L =$) and a list of the column headings for the table following. Data is only printed for those cells with either the horizontal, or vertical or both components of velocity non zero. The first column is the K or cell number in horizontal direction. The next two columns are ETA and PSI which are the horizontal and vertical components of the sum of old velocity, advection, body force and viscous terms from the difference approximation of the Navier-Stokes equations. These terms are primarily of diagnostic value. The next two columns are U and V and are the horizontal and vertical components of cell velocity as used in the calculations. The horizontal component of velocity is located at the right center edge of the computational cell, the vertical component of velocity is located at the top center edge of the computational cell. The next column is pressure, P, and is the total pressure at the center of the computational cell, including both dynamic and static pressure. The column X or R is the horizontal coordinate of the center of the cell. The last two columns are UPART and VPART, these are the average values of the horizontal and vertical components of the velocities of all the marker particles in that cell.

The next section of data edited is optional and its inclusion is dependent on the flag set in optional data card 1. This section contains the values of flags for all the cells and moving boundary data. The row number ($L =$) and column headings are given at the top of the table of data, middle of Fig. 3. The first column is the K number or computational cell

number in the horizontal direction. The next thirteen columns grouped in four groups give the values of the flags used during the computations 0 is off (no) and 1 is on (yes) for the flags. The first three are FULL, SUR, EMP and indicate: FULL, the cell has fluid (marker particles) and is surrounded on all four sides by cells containing fluid; SUR, the cell has fluid but is adjacent to an empty cell; EMP, the cell contains no fluid (marker particles). The next group of four are OB, OK, ARB, COR these flags all have to do with the arbitrary boundaries, and indicate: OB, cell contains an arbitrary boundary; OK, can use relaxation boundary condition to calculate pressure in the cell; ARB, particles are within the boundary sensing distance from the boundary and calculate pressure assuming fluid is in contact with the boundary; COR, the cell may contain a small boundary segment but does not contain enough fluid to be treated as a normal boundary cell, it is blended with an adjacent cell for computations. The next group of three are BND, IN, OUT and have to do with the row of cells around the computational mesh which do not enter the calculations, these are the top, bottom, left and right most rows, these flags indicate: BND, cell is a mesh boundary cell; IN, it is an inflow boundary; OUT, it is an outflow boundary. The last group of three flags is GAS, EMPBND and SNC, they are: GAS, contains gas bubble; EMPBND, not used; and SNC, cell is surface cell and not a corner cell, used in one of the velocity routines. The next column NPT is the number of marker particles in the cell, does not include special surface particles. The column KMPR is a pointer giving the K number of the cell to go to when a cell containing a small amount of fluid and arbitrary boundary is merged with another cell. The next two columns are DNORX and DNORY, these are the direction numbers of the unit normal vector at the center of

a boundary segment in a cell, pointing into the fluid. The next two columns, PNORX and PNORY are the horizontal and vertical coordinates of the center of the boundary segment in that cell. The last two columns are UBND and VBND and these are the values of the horizontal and vertical components of the velocity of the boundary segment in that cell for moving boundary problem

These two sections of data are then repeated for each row of the computational mesh until completed. Between edits, diagnostic information is printed out for each cycle and is shown at the bottom of Fig. 3. These data are: cycle number; the actual ratio of norm of the residual of pressure to norm of pressure ($ERROR =$), the limit for this is given on optional data card 2; BFLUX is the sum the dot products of the fluid velocity with respect to the boundary segments and the unit normal vector to the boundary segment, it is a measure of the fluid crossing the boundary and is analogous to the divergence; the problem time for the cycle ($TIME =$); the time step for the cycle ($DT =$); and the number of iterations required to relax the pressure iteration equations to within the limit given by $ERROR =$, there is a limit set in the routine of 200 iterations. If the relaxation equations cannot be satisfied within 200 iterations the problem automatically stops and edits everything.

More detailed information is given by Viecelli (1971) and should be used to gain a feel for the mechanics of the computational method.

Acknowledgement

The authors are most appreciative of the work performed by Nancy Willard in preparing the pages of listings and flow charts for printing.

This work was performed under the auspices of the U. S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

```

180X U96 SUP25R BOX U96
SURF25 BOUNDARY FREQUENCY NO. = 0.4 DISPLACEMENT / DEPTH = 0.5
150 40 0 1 0
3.60000E-04 6.00000E-02
3.22000E+01 0. 1.00000E-05 1.94000E+00 1.94000E+00
0. 0. 0.
1 150 6.00000E-02
-0 -0 -0.
1 40 6.00000E-02
-0 -0 -0.
3 3 25 2 10
1.00000E-01 0.
1.44000E+00 0.
1.44000E+00 5.00000E-01
1.00000E-01 5.00000E-01
1.00000E-01 0.
2 25 149 2 10
1.44000E+00 0.
8.88000E+00 0.
8.88000E+00 5.00000E-01
1.44000E+00 5.00000E-01
1.44000E+00 0.
-0 -0 -0 -0 -0 -0
1 150 1 40 0. 0. 0
-0 -0 -0 -0 -0 -0
-0 -0 -0. -0. -0. -0.
1.00000E-01 2.28100E+00 -0. -0. -0. -0.
1.00000E-01 -1.00000E-03 1
1 2
5 0. 2.88000E+00 0. 2.28000E+00 9.00000E-01 3.00000E-01
1 150 1 40
BOUNDARY ACTION N TW(N) XW(N) UW(N)
1 0. 0. 1.60565E+00
2 1.55700E-01 2.50000E-01 0.
5.00000E-01
1 6.00000E-03 1.00000E-01 1.00000E-01 1.00000E+00 -0. -0.
2 6.00000E-02 8.00000E-04 4.10000E-01 0. 3.60000E-04 -0.
3 5.00000E+02 0. 0. 5.00000E+02 -0. -0.
-0 -0. -0. -0. -0. -0.
CYCLE 0 TIME 0. DT 6.000000E-03
SURFACE PARTICLES
1.100E-01 1.000E-02 1.100E-01 3.000E-02 1.100E-01 5.000E-02
1.100E-01 7.000E-02 1.100E-01 9.000E-02 1.100E-01 1.100E-01
1.100E-01 1.700E-01 1.100E-01 1.500E-01 1.100E-01 1.700E-01
1.100E-01 1.100E-01 1.100E-01 0.100E-01 1.100E-01 2.800E-01

```

Fig. 1 Example 1, Sample Edit of Input Data for Problem Without Surface Particles.

```

180X U96 ME7A B0X U96
ME7A MENDICING ESCAPEMENT MOTION TRAIL PROBLEM 7.
252 18 0 1
2.25000E+04 3.00000E+02 1.00000E+06 1.03000E+03 1.03000E+03
3.80700E+00 0. 0.
1 252 3.00000E+02
-0 -0 -0.
1 18 3.00000E+02
-0 -0 -0.
2 2 35 5 6
0. 1.10000E+03
1.02000E+04 1.10000E+03
1.02000E+04 1.50000E+03
0. 1.50000E+03
0. 1.10000E+03
1 2 35 7 14
0. 1.50000E+03
1.02000E+04 1.50000E+03
1.02000E+04 3.90000E+03
0. 3.90000E+03
0. 1.50000E+03
2 2 252 18 16
0. 3.90000E+03
7.50000E+04 3.90000E+03
7.50000E+04 4.30000E+03
0. 4.30000E+03
0. 3.90000E+03
2 2 131 2 14
1.02000E+04 1.10000E+03
2.50000E+04 1.10000E+03
2.91000E+04 3.10000E+03
2.94000E+04 3.10000E+03
3.88000E+04 0.
3.90000E+04 0.
3.90000E+04 3.90000E+03
1.02000E+04 3.90000E+03
1.02000E+04 1.10000E+03
1 132 251 2 14
3.90000E+04 0.
7.50000E+04 0.
7.50000E+04 3.90000E+03
3.90000E+04 3.90000E+03
3.90000E+04 0.
-0 -0 -0 -0 -0 -0
1 252 1 18 0. 0. 0.
-0 -0 -0 -0 -0 -0
-0 -0 -0 -0 -0 -0
-0 -0 -0 -0 -0 -0
7.49500E+04 4.30000E+03 7.50000E+01 -0 -0 -0
2.50000E+01 4.30000E+03 7.50000E+01 1
-1.00000E+01 1.10000E+03 -0
2.50000E+04 1.10000E+03 -0
2.91000E+04 3.10000E+03 -0
2.94000E+04 3.10000E+03 -0
3.88000E+04 -3.87500E+00 1
2 5
5 0. 7.50000E+04 4.25000E+03 4.55000E+03 9.00000E-01 9.00000E-01
1 252 1 18
INTERIOR PARTICLES = 6954 SURFACE PARTICLES = 1000
BOUNDARY MOTION N TWIN X(N) UN(N)
1 0. 0. 2.00000E+00
2 5.00000E+00 1.00000E+01 0.
4.30000E+03
F31SEF31SEF315
1 1.00000E+00 1.00000E+01 1.00000E+01 0. -0. -0.
2 3.00000E+02 1.00000E-04 4.50000E-01 0. 2.25000E+04 -0.
3 1.50000E+02 0. 1.50000E+02 1.50000E+02 -0. -0.
-0 -0 -0 -0 -0 -0
CYCLE 0 TIME 0. DT 1.000000E+00
SURFACE PARTICLES
7.4950E+04 4.3000E+03 7.4875E+04 4.3000E+03 7.4800E+04 4.3000E+03
7.4725E+04 4.3000E+03 7.4650E+04 4.3000E+03 7.4575E+04 4.3000E+03
7.4500E+04 4.3000E+03 7.4425E+04 4.3000E+03 7.4350E+04 4.3000E+03
7.4275E+04 4.3000E+03 7.4200E+04 4.3000E+03 7.4125E+04 4.3000E+03
7.4050E+04 4.3000E+03 7.3975E+04 4.3000E+03 7.3900E+04 4.3000E+03
7.3825E+04 4.3000E+03 7.3750E+04 4.3000E+03 7.3675E+04 4.3000E+03

```

Fig. 2 Example 2, Sample Edit of Input Data for Problem With Surface Particles.

SUP25 BOUNDARY ROUTE NO. = 0.4 DISPLACEMENT / DEPTH = 0.5

2= 1.500000E-01 CYCLE= 3 TIME= 1.428742E-02

L= 4 K>ETA>PSI>U>V>F>X OR R, UPART>VPART

3	3.036270E+00	1.127355E+00	1.608952E+00	0.	0.	9.000000E-02	0.	0.
4	2.638123E+00	1.124748E+00	1.441210E+00	7.009392E-01	0.	1.500000E-01	1.555555E+00	3.553635E+00
5	2.254860E+00	1.071076E+00	1.250176E+00	5.471334E-01	5.835281E+01	2.100000E-01	1.357583E+00	5.717829E-01
6	1.906794E+00	9.815641E-01	1.065976E+00	5.842138E-01	5.388099E+01	2.700000E-01	1.148456E+00	4.907165E-01
7	1.601367E+00	8.740154E-01	9.000148E-01	5.153845E-01	4.965265E+01	3.300000E-01	9.635552E-01	4.334965E-01
8	1.338995E+00	7.621005E-01	7.555328E-01	4.472959E-01	4.585785E+01	3.900000E-01	8.126322E-01	3.754669E-01
9	1.116582E+00	6.346198E-01	6.321465E-01	3.836431E-01	4.253241E+01	4.500000E-01	6.823527E-01	3.227573E-01
10	9.295406E-01	5.563437E-01	5.278337E-01	3.262481E-01	3.965276E+01	5.100000E-01	5.713908E-01	2.746109E-01

149 -3.999946E-13 -1.258830E-04 5.228119E-13 -1.145887E-06 2.251015E+01 8.790000E+00 1.176573E-12 -1.074739E-06

149 0. -1.258830E-04 0. -1.145887E-06 2.251015E+01 8.850000E+00 3.487818E-13 -1.074739E-06

150 0. 0. -5.228119E-13 -1.145887E-06 2.251015E+01 8.910000E+00 0. 0.

L= 4 K>FULL>SUR>EMP>CB>OK>ARB>CGR>BND>IN>OUT>GAS>EMP>BND>SNC>NPT>KMP>IN>OK>IN>RY>PN>OK>PN>RY>USEID>VEND

1	000 0000 100 000	0 0 0.	0.	0.	0.	0.	0.
3	001 0001 000 000	0 0 0.	0.	0.	0.	0.	0.
4	010 1110 000 000	8 0 1.000000E+00	0.	1.229406E-01	1.500000E-01	1.605652E+00	0.
5	100 0010 000 000	11 0 0.	0.	0.	0.	0.	0.
6	100 0010 000 000	10 0 0.	0.	0.	0.	0.	0.
7	100 0010 000 000	9 0 0.	0.	0.	0.	0.	0.
8	100 0010 000 000	9 0 0.	0.	0.	0.	0.	0.
9	100 0010 000 000	9 0 0.	0.	0.	0.	0.	0.
10	100 0010 000 000	9 0 0.	0.	0.	0.	0.	0.

CYCLE 4 ERROR= 7.683E-04 BFLUX= 1.591E-02 TIME= 1.429E-02 DT= 2.516E-03 N= 58

CYCLE 5 ERROR= 6.657E-04 BFLUX= 1.282E-02 TIME= 1.680E-02 DT= 2.768E-03 N= 5

CYCLE 6 ERROR= 6.384E-04 BFLUX= 9.784E-03 TIME= 1.957E-02 DT= 3.045E-03 N= 2

CYCLE 7 ERROR= 5.754E-04 BFLUX= 8.150E-03 TIME= 2.262E-02 DT= 3.349E-03 N= 3

CYCLE 8 ERROR= 6.039E-04 BFLUX= 4.398E-03 TIME= 2.596E-02 DT= 3.684E-03 N= 6

CYCLE 9 ERROR= 6.199E-04 BFLUX= 6.739E-03 TIME= 2.965E-02 DT= 4.052E-03 N= 2

CYCLE 10 ERROR= 6.572E-04 BFLUX= 1.337E-02 TIME= 3.370E-02 DT= 4.459E-03 N= 7

CYCLE 11 ERROR= 7.609E-04 BFLUX= 1.565E-02 TIME= 3.816E-02 DT= 4.903E-03 N= 2

Fig. 3 Sample Edit of Computational Cell Parameters and Diagnostic Information.

APPENDIX D: FORTRAN LISTING

[illegible]

```

*      LIST8
*      FORTRAN                      SUBMAIN
      SUBROUTINE MACMAIN
      CODE ANALYSIS
      USE CMAC
      IA=3 $ IB=3 $ OPE=0.
      INTEGER EMPT,EMPM,EMPR
      GENERIC Q8QFLA

      ARBITRARY BOUNDARY MODIFICATION OF MARKER PARTICLE
      INCOMPRESSIBLE HYDRODYNAMIC CODE
      REF, LOS ALAMOS LABORATORY REPORT LA-3425 (1966)
      LAWRENCE RADIATION LABORATORY REPORT UCRL 71595 (1969)
      JOURNAL OF COMPUTATIONAL PHYSICS, VOL 4 NO. 4 (1969) P. 543

      FLOW IN, FLOW OUT, AND RIGID WALL BOUNDARIES ALLOWED AT KMAX, LMAX
      FLOW OUT, AND RIGID WALL BOUNDARIES ALLOWED AT K=1, L=1
      PLANE FLOW OR CYLINDRICAL SYMMETRY (ABOUT K LINE 1)

      RIT 2,360,NIXE,ID1,ID2,ID3 $ IGOB(36)=ID1 $ IGOB(37)=ID2
360  FORMAT(110,3A10) $ IGOB(38)=ID3
      CALL CRTID(2HJV,1,1)

      NTAL=0
      GO TO (361,362),NIXE
361  CALL GEN
      NHC, NHD, NHE ARE TAPE LABLES
      RIT 2,430,NHC,NHD,NHE
430  FORMAT(3R5)
      IF(NHC)431,367,431
431  NTP1=1
      NTP2=0
      NTAPE=NHC
      NTPE=1
      GO TO 367
362  CALL RDX
367  READ INPUT TAPE 2,363,MP,XXA,XXB,XXC,XXD,XXE,XXF
363  FORMAT(110,6E10,4)
      PRINT 363,MP,XXA,XXB,XXC,XXD,XXE,XXF
      IF(MP)418,418,364
364  GO TO(441,442,443,444,445,446),MP
441  DTR = XXA
      EDDT=XXB
      EDTIM=XXC
      OPE=XXD
      VPOP=XXE
      GO TO 367
442  EPS1=XXA
      EPS2=XXB
      BETA=XXC

```

```

      NGEN=XXD
      SMIN=XXE
      GO TO 367
443  NSTOP=XXA
      NLIMIT=XXB
      NEDIT=XXC
      NEDTM=XXD
      GO TO 367
444  DBUGA=XXA
      DBUGB=XXB
      DBUGC=XXC
      DDUOD=XXD
      GO TO 367
445  KDMN=XXA
      KDMX=XXB
      LDMN=XXC
      LDMX=XXD
      GO TO 367
446  CRTRA=XXA
      CRTRB=XXB
      CRTZA=XXC
      CRTZB=XXD
      XMAX=XXE
      YMAX=XXF
      GO TO 367
418  CONTINUE
      KMAXZ=KMAX-1
      LMAXZ=LMAX-1
      KMAXZZ=KMAX-2
      LMAXZZ=LMAX-2
      NGRID=KMAX*LMAX
      DTMAX=DR(1)**2*DZ(1)**2/(DR(1)**2+DZ(1)**2)
      DTMAX=.4*DTMAX/(MU+.0000001)*RHOA
      DTR = .5 * (DTR+DTMAX) - .5 * ABSF(DTR - DTMAX)
      DTMAX=DTR
      IF (NCYCLE) , ,B10
      DT = DTR
B10  RELAX=BETA*RHOA*DR(1)**2/DT
      IF(O)22,22,23
      22  CRAX=RHOA
          GO TO 24
      23  CRAX=G*Z(LMAX)
          CRAX=1.0/CRAX
      24  GG=G
          GO TO (32,33),NIXE
      32  CALL EDIT
      33  CONTINUE
0
C
C
C
C
      253  CONTINUE

      COMPUTATION OF THE TRANSPORT TERMS IN THE MOMENTUM EQUATION
      STORE IN ETA AND PSI
      UM=MU*DT
      DO 74 N=1,NGRID
      PSI(N)=0.0
      ETA(N)=0.0
      74  CONTINUE

```

CCC

COMPUTE HORIZONTAL TRANSPORT OF U MOMENTUM

```

DO 40 L=2,LMAXZ
KMM=(L-1)*KMAX
IF(NGEN)11,12
11 UR=.25*(U(KMM+1)+U(KMM+2))*2*RHOA      *R(2)**NEXP
GO TO 43
12 UBAR=.5*(U(KMM+1)+U(KMM+2))*RHOA      *R(2)**NEXP
IF(UBAR)41,42,42
42 UR=UBAR*U(KMM+1)
GO TO 43
41 UR=UBAR*U(KMM+2)
DO 40 K=2,KMAXZZ
KM=K+KMM
UL=UR
IF(NGEN)13,14
13 UR=.25*(U(KM)+U(KM+1))*2*RHOA      *R(K+1)**NEXP
GO TO 46
14 UBAR=.5*(U(KM)+U(KM+1))*RHOA      *R(K+1)**NEXP
IF(UBAR)44,45,45
44 UR=UBAR*U(KM+1)
GO TO 46
45 UR=UBAR*U(KM)
46 IF((.NOT.EMP(KM)).AND.(.NOT.EMP(KM+1)))47,40
47 ETA(KM)=ETA(KM)+RHOA*U(KM)
ETA(KM)=ETA(KM)-DT*(UR-UL)/(DRB(K)*RB(K)**NEXP)
40 CONTINUE

```

CCC

COMPUTE VERTICAL TRANSPORT OF U MOMENTUM

```

DO 48 K=2,KMAXZZ
VT=0.
DO 48 L=2,LMAXZ
KT=K+L*KMAX
KM=KT-KMAX
KB=KM-KMAX
VB=VT
VBAR=.5*(V(KM)+V(KM+1))
EMPT=EMP(KT).AND.EMP(KT+1)
EMPM=EMP(KM).AND.EMP(KM+1)
NEMPT=(.NOT.EMP(KT)).AND.(.NOT.EMP(KT+1))
NEMPM=(.NOT.EMP(KM)).AND.(.NOT.EMP(KM+1))
IF(EMPM)49,50
49 IF(NEMPT)52,51
51 VT=0.
GO TO 48
52 VT=RHOA*U(KT)*VBAR
GO TO 48
50 IF(NEMPM)76,25
25 IF(NEMPT)26,48
26 IF(VBAR)27,28,28
27 VT=RHOA*U(KT)*VBAR
GO TO 48
28 VT=RHOA*VBAR*U(KM)
GO TO 48
76 IF(VBAR)53,54,54
53 IF(EMPT)54,55
55 VT=RHOA*VBAR*U(KT)
GO TO 56

```

```

54 VT=RHOA*VBAR*U(KM)
56 IF(OUT(KB),OR,OUT(KT))48,82
82 ETA(KM)=ETA(KM)-DT*(VT-VB)/DZ(L)
48 CONTINUE

C
C C      COMPUTE VERTICAL TRANSPORT OF V MOMENTUM
      DO 57 K=2,KMAXZ
      IF(NGEN)15,16
15 VT=.25*(V(K)+V(K+KMAX))*2*RHOA
      GO TO 60
10 VBAR=.5*(V(K)+V(K+KMAX))*RHOA
      IF(VBAR)58,59,59
58 VT=VBAR*V(K+KMAX)
      GO TO 60
59 VT=VBAR*V(K)
60 DO 57 L=2,LMAXZZ
      KT=K+L*KMAX
      KM=KT-KMAX
      VB=VT
      IF(NGEN)17,18
17 VT=.25*(V(KM)+V(KT))*2*RHOA
      GO TO 64
18 VBAR=.5*(V(KM)+V(KT))*RHOA
      IF(VBAR)62,63,63
62 VT=VBAR*V(KT)
      GO TO 64
63 VT=VBAR*V(KM)
64 IF((.NOT.EMP(KM)).AND.(.NOT.EMP(KT)))61,57
61 PSI(KM)=PSI(KM)+RHOA*V(KM)
      PSI(KM)=PSI(KM)-DT*(VT-VB)/DZB(L)
57 CONTINUE

C
C C      COMPUTE HORIZONTAL TRANSPORT OF V MOMENTUM
      DO 65 L=2,LMAXZZ
      KMM=L*KMAX
      UR=0.
      DO 65 K=2,KMAXZ
      KT=K+KMM
      KM=KT-KMAX
      UL=UR
      UBAR=.5*(U(KM)+U(KT))*RB(K)**NEXP
      EMPR=EMP(KM+1).AND.EMP(KT+1)
      EMPM=EMP(KM).AND.EMP(KT)
      NEMPR=(.NOT.EMP(KM+1)).AND.(.NOT.EMP(KT+1))
      NEMPM=(.NOT.EMP(KM)).AND.(.NOT.EMP(KT))
      IF(EMPM)66,67
66 IF(NEMPR)69,68
68 UR=0.
      GO TO 65
69 UR=RHOA*UBAR*V(KM+1)
      GO TO 65
67 IF(NEMPM)75,29
29 IF(NEMPR)34,65
34 IF(UBAR)35,36,36
35 UR=RHOA*UBAR*V(KM+1)
      GO TO 65
36 UR=RHOA*UBAR*V(KM)
      GO TO 65

```

```

75 IF(UBAR)70,71,71
70 IF(EMPR)71,72
72 UR=RHOA*UBAR*V(KM+1)
GO TO 73
71 UR=RHOA*UBAR*V(KM)
73 IF(OUT(KM+1)).OR.OUT(KM-1))65,79
79 PSI(KM)=PSI(KM)-DT*(UR-UL)/(DR(K)*R(K)**NEXP)
65 CONTINUE

CCC
      COMPUTE VISCOUS TERMS
      IF(MU)80,80,81
81 DO 87 L=2,LMAXZ
      KTT=L*KMAX
      DO 87 K=2,KMAXZZ
      KT=K+KTT
      KM=KT-KMAX
      KB=KM-KMAX
      NEMPM=(.NOT.EMP(KM)).AND.(.NOT.EMP(KM+1))
      EMPT=EMP(KT).AND.EMP(KT+1)
      EMPB=EMP(KB).AND.EMP(KB+1)
      IF(NEMPM)88,87
88 IF(EMPB)89,90
89 UBOT=U(KM)
      GO TO 91
90 UBOT=U(KB)
91 IF(EMPT)92,93
92 UTOP=U(KM)
      GO TO 94
93 UTOP=U(KT)
94 ETA(KM)=ETA(KM)+UM*(UTOP+UBOT-2.*U(KM))/DZ(L)**2
      ETA(KM)=ETA(KM)-UM*(V(KM+1)-V(KB+1)-V(KM)+V(KB))/(DRB(K)*DZ(L))
87 CONTINUE
      DO 95 K=2,KMAXZ
      DO 95 L=2,LMAXZZ
      KT=K+L*KMAX
      KM=KT-KMAX
      NEMPM=(.NOT.EMP(KM)).AND.(.NOT.EMP(KT))
      EMPR=EMP(KM+1).AND.EMP(KT+1)
      EMPL=EMP(KM-1).AND.EMP(KT-1)
      IF(NEMPM)96,95
96 IF(EMPL)97,98
97 VLEFT=V(KM)
      GO TO 99
98 VLEFT=V(KM-1)
99 IF(EMPR)101,102
101 VRGHT=V(KM)
      GO TO 103
102 VRGHT=V(KM+1)
103 TEMPR=(U(KT)-U(KM))/DZB(L)-(VRGHT-V(KM))/DRB(K)
      TEMPL=(U(KT-1)-U(KM-1))/DZB(L)-(V(KM)-VLEFT)/DRB(K-1)
      PSI(KM)=PSI(KM)-UM*(TEMPR*RB(K)**NEXP-TEMPL*RB(K-1)**NEXP)
      1/(DR(K)*R(K)**NEXP)
95 CONTINUE
80 CONTINUE

CCCCC
      THIS COMPLETES THE COMPUTATION OF THE TRANSPORT TERMS

```

```

C
C
C 150 CONTINUE
C
C   GET NEW VELOCITIES AND PRESSURES
C   EPS=EPS2
C   CALL PRES (EPS)
C
C   GET NEW VELOCITIES AT FREE SURFACE-BOUNDARY INTERSECTION CELLS
C   CALL VEL57
C
C   MOVE THE PARTICLES
C   CALL TRAJ
C
C   TALLY NO. OF PARTICLES IN CELL
C   DO 351 N=1, NGRID
C   MPA=(MPRTA(N).SHL.50).SHR.50
C   NPRTA(N)=NPRTA(N)+MPA
351 CONTINUE
C
C   INPUT PARTICLES AT FLOW IN BOUNDARIES
C   CALL FLOWIN
C
C   RESET THE BYTE CELL FLAGS
C   CALL SHIFT
C   CALL REFLAG
C
C   NCYCLE=NCYCLE+1
C   TIME =TIME+DT
C
C   VMAX=0.0
C   DO C10 N=1, NGRID
C   UMAX=ABSF(U(N))+ABSF(V(N))
C10  VMAX=.5*(ABSF(VMAX-UMAX)+VMAX+UMAX)
C
C   COMPUTE NEW TIME STEP
C   IF(VMAX)415,415,416
C   DX = MIN1F(DR(1), UZ(1))
C   DTV = MIN1F(DX/2., EPS1) / VMAX
C   DT0 = MIN1F(DTV, DTMAX)
C   IF (DT - .9*DT0) , , A10
C   DT = 1.1 * DT
C   GO TO 415
C   A10 DT = DT0
C   415 CONTINUE
C   IF(DT-DTMAX)419,419,420
C   420 DT=DTMAX
C   419 CONTINUE
C   RELAX=BETA*RHOA*DR(1)**2/DT
C

```

```

C      CHECK CONSERVATION OF MASS
C
      PI=3.1415926535899
      TOTM=0.0
      DO 85 K=2,KMAXZ
      AREA=PI**NEXP*(RB(K)**NEXPP-RB(K-1)**NEXPP)
      DO 85 L=2,LMAXZ
      KM=K+(L-1)*KMAX
203    IF(FULL(KM).UN.SUR(KM))203,85
      TOTM=TOTM+RH0A *AREA*(ZB(L)-ZB(L-1))
      85 CONTINUE
      W0T 59,38,NCYCLE,TIME,TOTM $ CALL 000TBF
      38 FORMAT(14,2E9.2)
      PRINT 86,NCYCLE,TOTM,DT
      86 FORMAT(6H CYCL ,110,12H TOTAL MASS ,E20.9,4H DT ,E20.9)
      IF(NCYCLE-NSTOP)374,254,254
      374 CONTINUE
      IF(Q8QFLA(1))254,372
      372 IF(NCYCLE-NEDTM)373,375,375
      375 NEDTM=NEDTM+NEDIT
      GO TO 371
      373 IF(TIME-EDTIM)253,371,371
      371 CALL EDIT
      EDTIM=EDTIM+EDDT
      GO TO 253
      254 CALL EDIT
      IF(NHC)433,434,433
      433 CALL SAM
      434 CALL EXIT
      RETURN
      END

```



```

*      LIST8
*      FORTRAN          GEN
SUBROUTINE GEN
C      THIS ROUTINE IS THE GENERATOR FOR THE THE ARBITRARY BOUNDARY
C      VERSION THE THE MAC INCOMPRESSIBLE HYDRODYNAMIC CODE.
      USE CMAC
      USE PACK
      CODE ANALYSIS
      PI=3.14159265358979
      IA=3
      IB=3
      RIT 2,202,(MESS(1),I=1,7) $ WOT 3,202,(MESS(1),I=1,7)
202  FORMAT(7A10)
C
90  FORMAT(7E10.3)
4   FORMAT(7I10)
      REAL MU
      RIT 2,4,KMAX,LMAX,NEXP
      WOT 3,4,KMAX,LMAX,NEXP
      RIT 2,90,SMIN,EPS1,UBND,VBND
      WOT 3,86,SMIN,EPS1,UBND,VBND
      RIT 2,90,GZ,GR,MU,RHOA,RHOB
      WOT 3,86,GZ,GR,MU,RHOA,RHOB
      RIT 2,90,PAMB,PGASZ,GAMA
      WOT 3,86,PAMB,PGASZ,GAMA
      G=SQRT(GZ**2+GR**2)
      NEXPP=NEXP+1
      KMAXZ=KMAX-1
      KMAXZZ=KMAX-2
      LMAXZ=LMAX-1
      LMAXZZ=LMAX-2
      NCYCLE=0
      TIME=0.
      NDIV=0
      CUT=.0000001
      NGRID=KMAX*LMAX
      UM=MU
86  FORMAT(7E12.6)
87  FORMAT(7I13)
C
C      GENERATE COORDINATES
C      READ DR AND DZ
23  READ INPUT TAPE 2,20,M,N,XX
      IF(M)22,22,26
20  FORMAT(2I10,E10.3)
26  DO 21 K=M,N
      DR(K)=XX
21  CONTINUE
      GO TO 23
22  READ INPUT TAPE 2,20,M,N,XX
      IF(M)24,24,25
25  DO 27 L=M,N
      DZ(L)=XX
27  CONTINUE
      GO TO 22
24  CONTINUE
      RB(1)=1./10.0**10
      DO 28 K=1,KMAXZ
      RB(K+1)=RB(K)+DR(K+1)
28  CONTINUE

```

```

      ZB(1)=0.0
      DO 29 L=1,LMAXZ
      ZB(L+1)=ZB(L)+DZ(L+1)
29  CONTINUE
      R(1)=-.5*RB(2)
      Z(1)=-.5*ZB(2)
      DO 30 K=2,KMAX
      R(K)=(RB(K)+RB(K-1))*0.5
30  CONTINUE
      DO 31 L=2,LMAX
      Z(L)=(ZB(L)+ZB(L-1))*0.5
31  CONTINUE
      DO 70 K=1,KMAXZ
      DRB(K)=R(K+1)-R(K)
70  CONTINUE
      DO 71 L=1,LMAXZ
      DZB(L)=Z(L+1)-Z(L)
71  CONTINUE
C
      DO 39 K=1,NGRID
      P(K)=0.
      U(K)=0.
      V(K)=0.
      ETA(K)=0.
      PSI(K)=0.
C
C      MOMENTUM TRANSPORTS STORED IN ETA AND PSI.
C      FLAG CONTAINS CELL FLAG BYTES.  UAVG IS A PACKED WORD
C      CONTAINING SUM OF VELOCITY COMPONENTS OF INDIVIDUAL PARTICLES.
      UAVG(K)=0.
      FLAG(K)=0.
      DNORX(K)=0.
      DNORY(K)=0.
      PNORX(K)=0.
      PNORY(K)=0.
39  CONTINUE
      DO 40 L=2,LMAXZ
      KMM=(L-1)*KMAX
      DO 40 K=2,KMAXZ
      KM=K+KMM
      EMP(KM)=1
40  CONTINUE
C
C      READ IN A SET OF POINTS SPECIFYING A CLOSED CURVE LYING IN THE
C      REGION BOUNDED BY KA,KB,LA,LB.  PARTICLES ARE GENERATED AT DENSITY
C      NP**2 INSIDE THIS CURVE.  ANY NO. OF CLOSED REGIONS ALLOWED.
      N=0
32  RIT 2,33,NP,KA,KB,LA,LB
33  FORMAT(7I10)
      WOT 3,33,NP,KA,KB,LA,LB
      IF(NP)91,91,34
34  RIT 2,603,BPX(1),BPY(1)
      M=2
604  RIT 2,603,BPX(M),BPY(M)
      IF((BPX(1) .EQ. BPX(M)).INT.(BPY(1) .EQ. BPY(M)))605,606
606  M=M+1
      GO TO 604
603  FORMAT(2E10.3)
605  CONTINUE
      MMAX=M-1
      DO 35 L=LA,LB

```

```

KMM=(L-1)*KMAX
DO 35 K=KA,KB
  KM=K+KMM
  DKP=DR(K)/NP
  DLP=DZ(L)/NP
  RA=RB(K-1)-.5*DKP
  ZA=ZB(L-1)-.5*DLP
  DO 36 LP=1,NP
    ZZ=ZA+DLP*LP
    DO 36 KP=1,NP
      RR=RA+DKP*KP
      ANGLE=0.
      DO 600 J=1,MMAX
        X1=BPX(J)-RR
        X2=BPX(J+1)-RR
        Y1=BPY(J)-ZZ
        Y2=BPY(J+1)-ZZ
        CROSSP=X1*Y2-X2*Y1
        FMAG=(X1**2+Y1**2)*(X2**2+Y2**2)
        FMAG=SQRT1(FMAG)
        TEMP = CROSSP / FMAG
        IF (ABSF(TEMP) - 1.) A10, A10,
        TEMP = SIGNF(1., TEMP)
A10 ALPHA = ASINF(TEMP)
      DOT=X1*X2+Y1*Y2
      IF(DOT)607,608,608
607 IF(ALPHA)609,610,610
609 ANGLE=ANGLE-PI-ALPHA
      GO TO 600
610 ANGLE=ANGLE+PI-ALPHA
      GO TO 600
608 ANGLE=ANGLE+ALPHA
600 CONTINUE
      IF(ABSF(ANGLE)-.0001)36,612,612
612 CONTINUE
      N=N+1
      NPRTA(KM)=NPRTA(KM)+1
      RP(N)=LURP(N,RR)
      RP(N)=LLRP(N,ZZ)
      FULL(KM)=1
      EMP(KM)=0
36 CONTINUE
35 CONTINUE
  NDIV=N
  NPART=N
  GO TO 32

```

C
C
C
C

```

CELL VELOCITY INPUT, KIN=0, KOUT=0 FOR INTERIOR CELLS, FOR INFLOW
BOUNDARY SET KIN=1 FOR OUTFLOW BOUNDARY KOUT=1.
91 CONTINUE
614 RIT 2,613,KA,KB,LA,LB,UVEL,VVEL,KIN,KOUT
613 FORMAT(4I10,2E10.3,2I5)
  IF(KA)92,92,94
94 DO 95 L=LA,LB
  KMM=(L-1)*KMAX
  DO 95 K=KA,KB
    KM=K+KMM
    U(KM)=UVEL
    V(KM)=VVEL

```

```

704 IF(KIN)704,705
    UIN=UVEL
    VIN=VVEL
705 CONTINUE
    IN(KM)=KIN
    OUT(KM)=KOUT
95 CONTINUE
    GO TO 614
C
C SPECIFY LINE OF INPUT PARTICLES LYING JUST OUTSIDE INPUT BOUNDARY
C NBP=NO. OF PARTICLES. XXA, YYA = COORDINATES OF FIRST PARTICLE
C IN LINE, DDR AND DDZ GIVE SPACING
92 CONTINUE
    RIT 2,700,NBP,XXA,YYA,DDR,DDZ
700 FORMAT(15,4E12.4)
    IF(NBP)701,701,702
702 CONTINUE
    M=NBP+1
    NPART=NPART+NBP
    DO 706 N=NPART,M,-1
    RP(N)=RP(N-NBP)
706 CONTINUE
    RP(1)=LURP(1,XXA)
    RP(1)=LLRP(1,YYA)
    K = (XXA + DR(1)) / DR(1) + 1
    L=(YYA+DZ(1))/DZ(1)+1
    KM=K+(L-1)*KMAX
    NBC(1)=KM
    NPRTA(KM)=NPRTA(KM)+1
    DO 703 N=2,NBP
    XXA=XXA+DDR
    YYA=YYA+DDZ
    RP(N)=LURP(N,XXA)
    RP(N)=LLRP(N,YYA)
    K=(XXA+DR(1))/DR(1)+1
    L=(YYA+DZ(1))/DZ(1)+1
    KM=K+(L-1)*KMAX
    NPRTA(KM)=NPRTA(KM)+1
    NBC(N)=KM
703 CONTINUE
    NDIV=NPART
701 CONTINUE
C
C GENB LOADS ARBITRARILY SHAPED LAGRANGE BOUNDARIES CUTTING
C ACROSS EULERIAN MESH
C CALL GENB
C
C BYTE OB=1 IF CELL CONTAINS BOUNDARY SEGMENT. BYTE COR=1 IF CELL
C IS JUST OUTSIDE BOUNDARY OR CONTAINS SMALL PORTION OF BOUNDARY
C THAT CAN BE NEGLECTED
C SECTION SENSES CASES WHERE LAGRANGE BOUNDARY ENDS IN THE
C INTERIOR OF THE MESH (I.E. SEPARATED FLOW AT EDGE OF A PLATE)
C OR INTERSECTS THE BOUNDARY OF THE EULERIAN MESH
DO 408 L=2,LMAXZ
    KTT=L*KMAX
DO 408 K=2,KMAXZ
    KT=KTT+K
    KM=KT-KMAX
    KB=KM-KMAX
    ITFX=1

```

```

      IF(OB(KM)) 409, 408
409  IF(DNCRX(KM)) 410, 420, 411
410  TEST=(PTX(1SEC).EQ.RB(K)).UN.(PTX(1).EQ.RB(K))
      TEST=TEST.INT.ITFX
      IF(OB(KM+1)).UN.FULL(KM+1).UN.TEST) 420, 412
412  COR(KM+1)=1
      GO TO 420
411  TEST=(PTX(1SEC).EQ.RB(K-1)).UN.(PTX(1).EQ.RB(K-1))
      TEST=TEST.INT.ITFX
      IF(OB(KM-1)).UN.FULL(KM-1).UN.TEST) 420, 414
414  COR(KM-1)=1
420  IF(UNCRY(KM)) 421, 440, 422
421  TEST=(PTY(1SEC).EQ.ZB(L)).UN.(PTY(1).EQ.ZB(L))
      TEST=TEST.INT.ITFX
      IF(OB(KT)).UN.FULL(KT).UN.TEST) 440, 423
423  COR(KT)=1
      GO TO 440
422  TEST=(PTY(1SEC).EQ.ZB(L-1)).UN.(PTY(1).EQ.ZB(L))
      TEST=TEST.INT.ITFX
      IF(OB(KB)).UN.FULL(KB).UN.TEST) 440, 424
424  COR(KB)=1
440  CONTINUE
      IF(ITFX) 442, 408
442  CONTINUE
      TEST=COR(KB).UN.COR(KT).UN.COR(KM-1).UN.COR(KM+1)
      IF(OB(KM).INT.(.NOT.TEST)) 441, 408
441  ITFX=0
      GO TO 409
408  CONTINUE
C
C
C      CELLS CONTAINING CONVEX RIGHT ANGLE BOUNDARIES TREATED AS
      INTERIOR CELLS
      LOGICAL TA,TB,TC,TD
      DO 430 L=2,LMAXZ
      KTT=L*KMAX
      DO 430 K=2,KMAXZ
      KT=KTT+K
      KM=KT-KMAX
      KB=KM-KMAX
      TA=OB(KB).INT.OB(KM-1).INT.(.NOT.COR(KT)).INT.(.NOT.COR(KM+1))
      TB=OB(KB).INT.OB(KM+1).INT.(.NOT.COR(KM-1)).INT.(.NOT.COR(KT))
      TC=OB(KM+1).INT.OB(KT).INT.(.NOT.COR(KM-1)).INT.(.NOT.COR(KB))
      TD=OB(KM-1).INT.OB(KB).INT.(.NOT.COR(KM+1)).INT.(.NOT.COR(KT))
      TEST=OB(KM).INT.(TA.UN.TB.UN.TC.UN.TD)
      IF(TEST) 431, 430
431  OB(KM)=0
430  CONTINUE
O
      RIT 2, 381, ICRT, CRTRA, CRTRB, CRTZA, CRTZB, XMAX, YMAX
381  FORMAT(110, 6E10, 3)
      RIT 2, 382, KDMN, KDMX, LDMN, LDMX
382  FORMAT(6110)
C
      ITEST=0
      PGAS=PGASZ
      GASV=0.
      CALL REFLAG
      GASVZ=GASV
C
C      BYTE BND=1 FOR CELLS AT BOUNDARY OF EULERIAN MESH, EXCEPT

```

```

C      WHERE LAGRANGE BOUNDARY INTERSECTS IT. (FREES VELOCITY COMP. FOR
C      PRESSURE ITERATION).
      DO 56 L=2, LMAXZ
      KL=(L-1)*KMAX+1
      KR=L*KMAX
      IF(COR(KL))60,61
61     BND(KL)=1
60     IF(COR(KR))56,62
62     BND(KR)=1
56     CONTINUE
      DO 57 K=1, KMAX
      KT=K+LMAXZ*KMAX
      KTZ=K+(LMAXZ-1)*KMAX
      KB=K+KMAX
      IF(COR(K))63,64
64     BND(K)=1
63     IF(COR(KT))57,65
65     BND(KT)=1
57     CONTINUE
      IOZ=1
      NTAL=0
      RIT 2,382,LA,LB
      DO 72 L=LB,LA,-1 $ PR=GZ*RHOA*(Z(LB+1)-Z(L))
      DO 72 K=1,KMAX $ KM=(L-1)*KMAX+K $ P(KM)=PR
72     CONTINUE
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          GENB
SUBROUTINE GENB
C      THIS SUBROUTINE GENERATES THE NORMAL VECTORS DEFINING THE
C      ARBITRARY LAGRANGE BOUNDARIES.
C      DNORX AND DNORY ARE DIRECTION COSINES OF BOUNDARY NORMALS
C      PNORX AND PNORY ARE R AND Z COORDINATES OF BOUNDARY NORMALS
      USE CMAC
      USE PACK
      CODE ANALYSIS
      PI=3.14159265358979
      IA=3
      IB=3
      DEPSX=.05*DR(1)
      DEPSY=.05*DZ(1)
C
C      READ IN SEQUENCE OF POINTS SPECIFYING A CONTINUOUS SECTION
C      OF BOUNDARY. NSEQ NEGATIVE SKIP THIS ROUTINE.
C      IF NSEQ=0 CONTINUE READ, NSEQ=2 PROCESS BOUNDARY AND RETURN
C      TO 158, NSEQ=1 NO FURTHER SECTIONS TO BE READ - RETURN TO GEN
C      AFTER PROCESSING
158  N=1
120  RIT 2,121,XB(N),YB(N),NSEQ
121  FORMAT(2E10.4,I10)
      IF(NSEQ)157,123,122
123  N=N+1
      GO TO 120
122  CONTINUE
      NBMAX=N-1
C
      IGEO=0
      DO 124 I=1,NBMAX
C      FIND ALL INTERSECTIONS WITH R=CONST. MESH LINES.   STORE IN
C      IN RMP, ZMP. KLINE=NO. OF R INTERSECTIONS
      IF(XB(I+1)-XB(I))125,126,127
127  K1=(XB(I+1)-.000001)/DR(1)+2
      A=(YB(I+1)-YB(I))/(XB(I+1)-XB(I))
      KDK=1
      M=0
      K2=KMAX
      B=YB(I)-A*XB(I)
      DO 128 K=K1,K2,KDK
      IF(RB(K)-XB(I+1)-.000001)129,129,130
129  M=M+1
      RMP(M)=RB(K)
      ZMP(M)=A*RMP(M)+B
128  CONTINUE
130  KLINE=M
      GO TO 134
125  K1=(XB(I)-.000001)/DR(1)+1
      A=(YB(I+1)-YB(I))/(XB(I+1)-XB(I))
      K2=1
      B=YB(I)-A*XB(I)
      KDK=-1
      M=0
      DO 131 K=K1,K2,KDK
      IF(RB(K)-XB(I+1)+.000001)132,133,133
133  M=M+1
      RMP(M)=RB(K)
      ZMP(M)=A*RMP(M)+B

```

```

131 CONTINUE
132 KLINE=M
GO TO 134
126 KLINE=0
134 CONTINUE
C FIND ALL INTERSECTIONS WITH Z=CONST. MESH LINES. STORE IN
C RNP, ZNP, LLINE=NO. OF Z INTERSECTIONS.
IF(YB(I+1)-YB(I))135,136,137
137 L1=(YB(I)+.000001)/DZ(I)+2
A=(XB(I+1)-XB(I))/(YB(I+1)-YB(I))
L2=LMAX
B=XB(I)-A*YB(I)
LDL=1
N=0
DO 138 L=L1,L2,LDL
IF(ZB(L)-YB(I+1)-.000001)139,139,140
139 N=N+1
ZNP(N)=ZB(L)
RNP(N)=A*ZNP(N)+B
138 CONTINUE
140 LLINE=N
GO TO 144
135 L1=(YB(I)-.000001)/DZ(I)+1
A=(XB(I+1)-XB(I))/(YB(I+1)-YB(I))
L2=1
B=XB(I)-A*YB(I)
LDL=-1
N=0
DO 141 L=L1,L2,LDL
IF(ZB(L)-YB(I+1)+.000001)142,143,143
143 N=N+1
ZNP(N)=ZB(L)
RNP(N)=ZNP(N)*A+B
141 CONTINUE
142 LLINE=N
GO TO 144
136 LLINE=0
144 CONTINUE
C LOAD ALL INTERSECTIONS INTO RNP,ZMP AND PLACE POINTS IN ORDER OF
C INCREASING DISTANCE FROM THE START OF THE SEGMENT.
DO 145 J=1,LLINE
JJ=J+KLINE
RNP(JJ)=RNP(J)
ZMP(JJ)=ZNP(J)
145 CONTINUE
JMAX=KLINE+LLINE
DO 146 J=1,JMAX
DS(J)=(RNP(J)-XB(I))**2 +(ZMP(J)-YB(I))**2
146 CONTINUE
DO 147 N=1,JMAX
TEST=10.0**10
MTRANS=0
DO 148 M=1,JMAX
IF(DS(M)-TEST)149,149,148
149 TEST=DS(M)
MTRANS=M
148 CONTINUE
RNP(N)=RNP(MTRANS)
ZNP(N)=ZMP(MTRANS)
DS(MTRANS)=10.0**20

```



```

147 CONTINUE
C ACCUMULATE INTERSECTIONS FROM EACH SEGMENT IN PTX,PTY
  J1=ISEC +1
  J2=ISEC+JMAX
  N=0
  DO 150 J=J1,J2
    N=N+1
    PTX(J)=RNP(N)
    PTY(J)=ZNP(N)
150 CONTINUE
  ISEC=ISEC+JMAX
C ISEC=TOTAL NO. OF INTERSECTIONS OF LAGRANGE BOUNDARY SECTION
C WITH EULERIAN MESH.
124 CONTINUE
C
C CHECK FOR DUPLICATE POINTS.
  I=1
153 I=I+1
156 IF(ABSF(PTX(I)-PTX(I-1))-DEPSX)160,160,151
160 IF(ABSF(PTY(I)-PTY(I-1))-DEPSY)152,152,151
152 ISEC=ISEC-1
  DO 154 J=I,ISEC
    PTX(J)=PTX(J+1)
    PTY(J)=PTY(J+1)
154 CONTINUE
  IF(I-ISEC)156,156,155
151 IF(I-ISEC)153,155,155
155 CONTINUE
C
615 FORMAT(2E10,3)
C
C WHEN SECTION ENDS IN THE INTERIOR OF THE MESH REMOVE ANY
C SMALL END SEGMENTS.
  ISECZ=ISEC-1
  DEP=(PTX(ISEC)-PTX(ISECZ))*2+(PTY(ISEC)-PTY(ISECZ))*2
  DEP=SQRTF(DEP)
  IF(DEP-.2*DR(1))23,24,24
23 ISEC=ISECZ
24 CONTINUE
  DEP=(PTX(2)-PTX(1))*2+(PTY(2)-PTY(1))*2
  DEP=SQRTF(DEP)
  IF(DEP-.2*DR(1))21,22,22
21 I=1
  GO TO 501
C
C COMPUTE BOUNDARY NORMALS FOR EACH CELL CONTAINING A SEGMENT
C OF BOUNDARY AND SET CELL FLAGS. PLACE NORMALS AT MID POINT OF
C SEGMENTS. CONVENTION IS THAT NORMAL POINTS TO THE LEFT AS ONE
C MOVES FROM I TO I+1 ALONG SEGMENT.
22 I=0
501 I=I+1
C
C DETERMINE IF SEGMENT LIES ON TOP OF A MESH LINE.
  K=(.5*(PTX(I+1)+PTX(I)))/DR(1) +2
  L=(1.0000001*PTY(I))/DZ(1)+1
  IF(. (PTY(I).EQ.PTY(I+1)).INT. (PTY(I).EQ.ZB(L)))200,201
200 IF(PTX(I+1)-PTX(I))202,202,203
203 KM=L*KMAX +K
  KB=KM-KMAX
  DNORX(KM)=0.

```

```

      DNORX(KM)=1.
      PNORX(KM)=.5*(PTX(I)+PTX(I+1))
      PNORY(KM)=PTY(I)
      OB(KM)=1
      COR(KB)=1
      GO TO 617
202 KM=(L-1)*KMAX +K
      KT=KM+KMAX
      DNORX(KM)=0.
      DNORY(KM)=-1.
      PNORX(KM)=.5*(PTX(I)+PTX(I+1))
      PNORY(KM)=PTY(I)
      COR(KT)=1
      OB(KM)=1
      GO TO 617
201 L=(.5*(PTY(I+1)+PTY(I)))/DZ(1) +2
      K=(1.0000001*PTX(I))/DR(1)+1
      IF((PTX(I).EQ.PTX(I+1)).INT.(PTX(I).EQ.RB(K)))204,205
204 IF(PTY(I+1)-PTY(I))206,206,207
207 KM=(L-1)*KMAX +K -1
      DNORX(KM)=-1.
      DNORY(KM)=0.
      PNORX(KM)=PTX(I)
      PNORY(KM)=.5*(PTY(I+1)+PTY(I))
      OB(KM)=1
      COR(KM+1)=1
      GO TO 617
206 KM=(L-1)*KMAX +K
      DNORX(KM)=1.
      DNORY(KM)=0.
      PNORX(KM)=PTX(I)
      PNORY(KM)=.5*(PTY(I+1)+PTY(I))
      COR(KM-1)=1
      OB(KM)=1
617 CONTINUE
      GO TO 500

C
C      GENERAL CASE, CELL SEGMENT NOT ON MESH LINE. COMPUTE AREA
C      BOUNDED BY SEGMENT AND INTERIOR SIDES OF CELL. USE LINE INTEGRAL
C 205 CONTINUE
C      NOW DO THE PART OF THE LINE INTEGRAL THAT
C      COMES FROM THE BOUNDARY SEGMENT.
      DX=PTX(I+1)-PTX(I)
      XBAR=.5*(PTX(I)+PTX(I+1))
      DY=PTY(I+1)-PTY(I)
      YBAR=.5*(PTY(I)+PTY(I+1))
      K= XBAR/DR(1) +1
      L= YBAR/DZ(1) +1
      SUM=-YBAR*DX
      IF(PTX(I+1)-RB(K))209,208,209
C      COMPUTE THE REST OF THE INTEGRAL
208 IF(PTY(I)-ZB(L))210,211,210
210 SUM=SUM -ZB(L)*DR(1) -(PTX(I)-RB(K+1))*ZB(L+1)
      GO TO 300
211 SUM=SUM- (PTX(I)-RB(K))*ZB(L)
      GO TO 300
209 IF(PTY(I+1)-ZB(L))212,213,212
213 SUM=SUM -(RB(K+1)-PTX(I+1))*ZB(L) -(PTX(I)-RB(K+1))*ZB(L+1)
      GO TO 300
212 IF(PTX(I+1)-RB(K+1))214,215,214

```

```

215 IF(PTY(I)-ZB(L+1))216,217,216
216 SUM=SUM +ZB(L+1)*DR(1) -(PTX(I)-RB(K))*ZB(L)
GO TO 300
217 SUM=SUM -(PTX(I)-RB(K+1))*ZB(L+1)
GO TO 300
214 SUM=SUM -(RB(K)-PTX(I+1))*ZB(L+1) -(PTX(I)-RB(K))*ZB(L)
300 CONTINUE
C
C THE FLUID AREA OF THE CELL IS STORED IN SUM.
C NEXT, BLEND AND COMPUTE NORMALS
C IGNORE CELLS WITH AREA LESS THAN SMIN, EITHER REMOVE
C SEGMENT OR ASSOCIATE IT WITH A NEIGHBORING CELL.
KPRM=K+1
LPRM=L+1
IF(SUM-SMIN)301,400,400
C
C DETERMINE WHICH CELL NORMAL POINTS NEAREST TO. IF THE I-1 INTER-
C SECTION LIES ON A SIDE OF THAT CELL REMOVE THE I TH INTERSECTION
C IF THE I+2 INTERSECTION LIES ON THE CELL SIDE REMOVE THE I+1
C INTERSECTION.
C WHEN SEGMENT IS NEARLY PARALLEL TO THE R OR Z MESH LINES
C NEITHER CASE MAY OCCUR
301 IF(ABSF(DY)-ABSF(DX))302,302,303
303 IF(DY)304,304,305
304 KPRM=K+2
GO TO 399
305 KPRM=K
GO TO 399
302 IF(DX)306,306,307
306 LPRM=L
GO TO 399
307 LPRM=L+2
399 CONTINUE
KKM=L*KMAX+K+1
COR(KKM)=1
XAVG=.5*(PTX(I+2)+PTX(I+1))
YAVG=.5*(PTY(I+2)+PTY(I+1))
KAVG= XAVG/DR(1) +2
LAVG= YAVG/DZ(1) +2
IF( (KPRM.EQ.KAVG).INT.(LPRM.EQ.LAVG))308,309
308 ISEC=ISEC-1
IZ=I+1
DO 310 J=IZ,ISEC
PTX(J)=PTX(J+1)
PTY(J)=PTY(J+1)
310 CONTINUE
GO TO 399
309 XAVG=.5*(PTX(I)+PTX(I-1))
YAVG=.5*(PTY(I)+PTY(I-1))
KAVG= XAVG/DR(1) +2
LAVG= YAVG/DZ(1) +2
KCEL=(LAVG-1)*KMAX+KAVG & KOEL=KOEL+NPRTB(KCEL)
KMPT=(LPRM-1)*KMAX+KPRM
IF(KMPT.EQ.KCEL)311,400
311 ISEC=ISEC-1
IZ=I
I=I-1
DO 314 J=IZ,ISEC
PTX(J)=PTX(J+1)
PTY(J)=PTY(J+1)

```

```

314 CONTINUE
398 CONTINUE
    XBAR=.5*(PTX(I+1)+PTX(I))
    YBAR=.5*(PTY(I+1)+PTY(I))
    DX=PTX(I+1)-PTX(I)
    DY=PTY(I+1)-PTY(I)
400 CONTINUE
    FMAG=SQRTF(DX**2+DY**2)
    IF(FMAG)402,402,401
402 WOT 59,403
    CALL C00TBF
403 FORMAT(1X,6HFMA9=0 )
    CALL QUIT(1)
401 CONTINUE
    KM=(LPRM-1)*KMAX +KPRM
    KAVG=XBAR/DR(1)+2
    LAVG=YBAR/DZ(1)+2
    KBAR=(LAVG-1)*KMAX+KAVG
    KSTOR=KM-KBAR
    NPRTB(KBAR)=KSTOR
C    NPRTB IS A BYTE WORD (PART OF FLAG) CONTAINING A POINTER FOR THE
C    CELL WHICH CONTAINS THE MID POINT OF THE BOUNDARY SEGMENT
C    INDICATING THE INDEX OF THE CELL TO BE USED IN
C    CALCULATING THE BOUNDARY PRESSURE.
    CB(KM)=1
    DNORX(KM)=-DY/FMAG
    DNORY(KM)=DX/FMAG
    PNORX(KM)=XBAR
    PNORY(KM)=YBAR
500 IF(I-ISEC+1)501,502,502
502 CONTINUE
C
C    IF PARTICLES IN CELL ARE WITHIN EPS1 OF BOUNDARY SEGMENT
C    ASSOCIATED WITH CELL SET BYTE ARB=1
    DO 48 N=1,NPART
    XR=SRP(N)
    XZ=SLRP(N)
    L=XZ/DZ(1) +2
    K=XR/DR(1) +2
    KBAR=(L-1)*KMAX+K
    KSTOR=(NPRTB(KBAR).SHL.50).SHR.50
    KM=KBAR+KSTOR
    IF((DNORX(KM).NE.0).OR.(DNORY(KM).NE.0))49,48
49 PDOTN=(XR-PNORX(KM))*DNORX(KM) + (XZ-PNORY(KM))*DNORY(KM)
    IF(PDOTN-EPS1)50,50,48
50 ARB(KM)=1
48 CONTINUE
C
C    IF(NSEQ-2)157,158,158
157 CONTINUE
C
    RETURN
    END

```

```

*      LIST8
*      FORTRAN          PRES
      SUBROUTINE PRES (EPS)
      USE CMAC
      CODE ANALYSIS
      IB=3

C      THIS ROUTINE SOLVES POISSON EQUATION FOR THE PRESSURE
C      BY SUCCESSIVE OVER RELAXATION. NOTE,
C      U=ETA-DP/DR , V=PSI-DP/DZ
C      WHERE ETA , PSI ARE CONSTANTS DURING ITERATION
C      THEREFORE P(N+1)=P(N)-RELAX*DIVERGENCE(VELOCITY)
C      GENERATES SOR PROVIDED U AND V ARE RECOMPUTED EACH TIME THE
C      CELL PRESSURE IS ADJUSTED
C      RELAX PARAMETER HAS MAXIMUM LIMIT FOR STABILITY AND OPTIMUM
C      VALUE FOR MOST RAPID CONVERGENCE
C      RELAX MAX=.5*RHO*DX**2/DT
C      OPTIMUM VALUE IS A FUNCTION OF GEOMETRY ETC. USUALLY VERY CLOSE T
C      TO MAX VALUE. SEE ANY SOR TEXT.

      LOGICAL TEST
      LOGICAL TA,TB,TC,TV,TH
      NEX=0
      CALL VEL(2)
33      M=0
424      N=0
188      N=N+1
      ENORM=0.
      PNORM=0.
      DO 423 L=2,LMAXZ
      KTT=L*KMAX
      DO 423 K=2,KMAXZ
      KP=K
      LP=L
      NT=KTT+KP
      NM=NT-KMAX
      NB=NM-KMAX
      IF(FULL(NM).INT.(.NOT.OB(NM)))21,22
21      DIV=(U(NM)*RB(K)**NEXP-U(NM-1)*RB(K-1)**NEXP)/(R(K)**NEXP*DR(1))
      DIV=DIV+(V(NM)-V(NB))/DZ(1)
      GO TO 24

C      IN BOUNDARY CELLS REPLACE DIV(V) BY (BOUNDARY NORMAL) DOT
C      (VELOCITY OF PARTICLE AT MID POINT OF LAGRANGE BOUNDARY
C      SEGMENT) . USE USUAL MAC AREA WEIGHTED FORMULA TO COMPUTE THIS
C      VELOCITY
22      CONTINUE
      IF(OK(NM))23,422
23      DLLR=DR(KP)
      DLLZ=DZ(LP)
      IF(PNORY(NM)-Z(LP))52,52,53
52      LB=LP-1
      GO TO 69
53      CONTINUE
      LB=LP
69      KT= LB*KMAX+KP
      KM= (LB-1)*KMAX+KP
      SX=(R(KP)-PNORX(NM))/DLLR
      SY=(Z(LB)-PNORY(NM))/DLLZ
      IF(U(KT-1))54,55,54

```

```

55 U1=U(NM-1)
   GO TO 56
54 U1=U(KT-1)
56 IF(U(KT))57,58,57
58 U2=U(NM)
   GO TO 59
57 U2=U(KT)
59 IF(U(KM-1))60,61,60
61 U3=U(NM-1)
   GO TO 62
60 U3=U(KM-1)
62 IF(U(KM))63,64,63
64 U4=U(NM)
   GO TO 65
63 U4=U(KM)
65 UP=(.5+SX)*(.5-SY)*U1 + (.5-SX)*(.5-SY)*U2
   UP=UP + (.5+SX)*(.5+SY)*U3 + (.5-SX)*(.5+SY)*U4
   IF(PNORX(NM)-R(KP))66,66,67
66 KB=KP-1
   GO TO 68
67 KB=KP
68 KD=(LP-2)*KMAX+KB
   KM=(LP-1)*KMAX+KB
   SX=(RB(KB)-PNORX(NM))/DLLR
   SY=(Z(LP)-PNORY(NM))/DLLZ
   IF(V(KM))70,71,70
71 V1=V(NM)
   GO TO 72
70 V1=V(KM)
72 IF(V(KM+1))73,74,73
74 V2=V(NM)
   GO TO 75
73 V2=V(KM+1)
75 IF(V(KD))77,76,77
76 V3=V(NB)
   GO TO 78
77 V3=V(KD)
78 IF(V(KD+1))79,80,79
80 V4=V(NB)
   GO TO 81
79 V4=V(KD+1)
81 VP=(.5+SX)*(.5-SY)*V1 + (.5-SX)*(.5-SY)*V2
   VP=VP + (.5+SX)*(.5+SY)*V3 + (.5-SX)*(.5+SY)*V4
   UP=UP-UBND
   VP=VP-VBND
   DOT=DNORX(NM)*UP + DNORY(NM)*VP
   DIV=DOT/DR(KP)
24 TEMP=RELAX*DIV
   P(NM)=P(NM)-TEMP
C
C   IF P NEGATIVE IN BOUNDARY CELL SET P=0 TO ALLOW FLOW SEPARATION.
93 IF(P(NM))91,92,92
91 P(NM)=0.
   TEMP=0.
C
C   FORM NORM**2 OF RESIDUALS AND PRESSURE.
92 ENORM=ENORM+TEMP**2
   PNORM=PNORM+P(NM)**2
422 CALL VELP(1,NM,K,L)
423 CONTINUE
C

```

```

C      APPLY BOUNDARY CONDITIONS
DO 183 L=2,LMAXZ
KL=2+(L-1)*KMAX
P(KL-1)=P(KL)-GR*RHOA      *(R(2)-R(1))
183 CONTINUE
DO 20 L=2,LMAXZZ
KR=L*KMAX
P(KR)=P(KR-1)+GR*RHOA      *(R(KMAX)-R(KMAXZ))
20 CONTINUE
C      REFLECT PRESSURES AT TOP AND BOTTOM BOUNDARIES
DO 184 K=2,KMAX7
KT=K+(LMAXZ-1)*KMAX
KTT=K+LMAXZ*KMAX
KB=K+KMAX
KBB=K
P(KTT)=P(KT)-GZ*RHOA      *(Z(LMAX)-Z(LMAXZ))
P(KBB)=P(KB)+GZ*RHOA      *(Z(2)-Z(1))
184 CONTINUE
CALL VEL(1)

C
C      TEST CONVERGENCE BY COMPARING NORMS.
ERROR=SQRT((ENORM/(PNORM+.0000001)))
C      WRITE OUT ERROR ON TTY FOR DEBUGGING
425 IF(DEBUGB-1.)405,405,406
406 WRITE OUTPUT TAPE 1B,187,ERROR
187 FORMAT(7HERROR=,E20.8)
405 IF(ERROR-EPS)189,426,426
426 IF(N-400)188,427,427
427 WOT 1B,428,ERROR
PRINT 428,ERROR
428 FORMAT(5X5HERROR,E18.6)
189 CONTINUE
RETURN
END

```

```

*      LIST8
*      FORTRAN          TRAJ
      SUBROUTINE TRAJ
      USE CMAC
      USE PACK
      CODE ANALYSIS

C      THIS ROUTINE MOVES THE MARKER PARTICLES BY USUAL MAC
C      WEIGHTING FORMULA.  R AND Z COORDINATES OF PARTICLE PACKED IN RP.
C
      IB=3
      IA=3
      DO 317 N=1, NGRID
      UAVG(N)=0.0
      MFRTA(N)=0
317  CONTINUE
      N=0
436  N=N+1
449  CONTINUE
      IF(N-NPART) 450, 450, 437
450  CONTINUE
C      UNPACK RP
      XR=SRP(N)
      XZ=SLRP(N)
      KP=(XR+DR(1))/DR(1)+1
      LP=(XZ+DZ(1))/DZ(1)+1
      NT= LP*KMAX+KP
      NM=NT-KMAX
      NB=NM-KMAX
      DLLR=RB(KP)-RB(KP-1)
      DLLZ=ZB(LP)-ZB(LP-1)
      IF( XZ -Z(LP)) 52, 52, 53
52  LB=LP-1
      GO TO 69
53  CCNTINUE
      LB=LP
69  KT= LB*KMAX+KP
      KM= (LB-1)*KMAX+KP
      SX=(R(KP)- XR )/DLLR
      SY=(ZB(LB)- XZ )/DLLZ
      IF(U(KT-1)) 54, 55, 54
55  U1=U(NM-1)
      GO TO 56
54  U1=U(KT-1)
56  IF(U(KT)) 57, 58, 57
58  U2=U(NM)
      GO TO 59
57  U2=U(KT)
59  IF(U(KM-1)) 60, 61, 60
61  U3=U(NM-1)
      GO TO 62
60  U3=U(KM-1)
62  IF(U(KM)) 63, 64, 63
64  U4=U(NM)
      GO TO 65
63  U4=U(KM)
65  UP=(.5+SX)*(.5-SY)*U1 + (.5-SX)*(.5-SY)*U2
      UP=UP + (.5+SX)*(.5+SY)*U3 + (.5-SX)*(.5+SY)*U4
      RAD= XR +UP*DT
      IF( XR -R(KP)) 66, 66, 67

```



```

66 KB=KP-1
GO TO 68
67 KB=KP
68 KD=(LP-2)*KMAX+KB
KM=(LP-1)*KMAX+KB
SX=(RB(KB)-XR)/DLLR
SY=(Z(LP)-XZ)/DLLZ
IF(V(KM))70,71,70
71 V1=V(NM)
GO TO 72
70 V1=V(KM)
72 IF(V(KM+1))73,74,73
74 V2=V(NM)
GO TO 75
73 V2=V(KM+1)
75 IF(V(KD))77,76,77
76 V3=V(NB)
GO TO 78
77 V3=V(KD)
78 IF(V(KD+1))79,80,79
80 V4=V(NB)
GO TO 81
79 V4=V(KD+1)
81 VP=(.5+SX)*(.5-SY)*V1 + (.5-SX)*(.5-SY)*V2
VP=VP + (.5+SX)*(.5+SY)*V3 + (.5-SX)*(.5+SY)*V4
ZALT=XZ +VP*DT

FINISHED COMPUTING NEW COORDINATES OF PARTICLE
NOW DETERMINE IF PARTICLE HAS CROSSED CELL BOUNDARY

RAD=RAD.INT.MASKL
ZALT=ZALT.INT.MASKL
KK=(RAD+DR(1))/DR(1)+1
LL=(ZALT+DZ(1))/DZ(1)+1
FAR = 0.
JM = 1
IF (KK - 1) A40, A10,
IF (KK - KMAX) A20, ,A10
A10 FAR = 1.
A20 IF (LL - 1) A40, A30,
IF (LL - LMAX) ,A30, A40
IF (FAR) 346, 346,
A30 JM = KK + (LL-1) * KMAX
IF (IN(JM)) 346,
IF (OUT(JM)) A50, A40
A40 PRINT W1, UP, VP, XR, XZ, RAD, ZALT
W1 FORMAT (4H UP=, E10.3, 4H VP=, E10.3, 4H XR=, E10.3, 4H XZ=,E10.3,
1 5H RAD=, E10.3, 6H ZALT=, E10.3)
A50 MPA = (MPRTA(NM) .SHL. 50) .SHR. 50
MPRTA(NM) = MPA - 1
REMOVE PARTICLES OUTSIDE MESH BOUNDARY.
45 RP(N)=RP(NPART)
NPART=NPART-1
NDIV=NPART
MPRTA(JM)=0
GO TO 449
346 JM=KK+(LL-1)*KMAX

```

```

C      SUM VELOCITIES OF PARTICLES IN CELL
C      PACK U IN UAVG UPPER
C      PACK V IN UAVG LOWER
C
      J=JM
      UBAR=SUUA(J)
      TEMP=UP+UBAR
      UAVG(J)=LUUA(J,TEMP)
      VBAR=SLUA(J)
      TEMP=VP+VBAR
      UAVG(J)=LLUA(J,TEMP)
      IF(JM-NM)89, 46, 89
89     MPRTA(JM)=1+((MPRTA(JM),SHL,50),SHR,50)
98     MPA=(MPRTA(NM),SHL,50),SHR,50
      MPRTA(NM)=MPA-1
C      PACK RP
      46 RP(N)=LURP(N,RAD)
      RP(N)=LLRP(N,ZALT)
      GO TO 436
437 CONTINUE
C
C      TALLY OF GAINS AND LOSSES OF PARTICLES STORED IN BYTE WORD MPRTA
C
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          FLOW
      SUBROUTINE FLOWIN
      USE CMAC
      USE PACK
      CODE ANALYSIS
C
C      THIS ROUTINE GENERATES NEW INFLOW PARTICLES AS THE OLD ONES MOVE
C      INSIDE THE BOUNDARY. NEW PARTICLE COORDINATES ALWAYS STORED
C      AT BEGINING OF RP. OLD ONES SHIFTED TO THE END.
      DO 1 N=1,NBP
      L=NBC(N)
      X=SRP(N)
      Y=SLRP(N)
      KK=(X+DR(1))/DR(1)+1
      LL=(Y+DZ(1))/DZ(1)+1
      KM=KK+(LL-1)*KMAX
      IF(KM-L)2,1,3
2     IF(KM-L+1)4,5,1
5     X=X+DKP
      GO TO 6
4     Y=Y+DLP
      GO TO 6
3     IF(KM-L-1)1,7,8
7     X=X-DKP
      GO TO 6
8     Y=Y-DLP
6     NPART=NPART+1
      RP(NPART)=RP(N)
      NDIV=NPART
      RP(N)=LURP(N,X)
      RP(N)=LLRP(N,Y)
      NPRTA(L)=NPRTA(L)+1
1     CONTINUE
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          SHIFT
*      SUBROUTINE SHIFT
*      USE CMAC
*      USE PACK
*      CODE ANALYSIS
C
C      WHEN FLUID PARTICLES COME WITHIN EPS1 OF BOUNDARY SET ARB=1
C
220 DO 48 N=1, NPART
    XR=SRP(N)
    XZ=SLRP(N)
    L=XZ/DZ(1) +2
    K=XR/DR(1) +2
    KBAR=(L-1)*KMAX+K
    KSTOR=(NPRTB(KBAR).SHL.50).SHR.50
    KM=KBAR+KSTOR
    IF((DNORX(KM).NE.0).OR.(DNORY(KM).NE.0))49,48
49  PDOTN=(XR-PNORX(KM))*DNORX(KM) + (XZ-PNORY(KM))*DNORY(KM)
50  IF(PDOTN-EPS1)50,50,48
50  ARB(KM)=1
48  CONTINUE
C
      RETURN
      END

```

```

*      LIST0
*      FORTRAN          REFLAG
      SUBROUTINE REFLAG
      USE CMAC
      USE PACK

C      THIS ROUTINE RESETS THE BYTE CELL FLAGS.  FREE SURFACE
C      TREATED ESSENTIALLY THE SAME AS IN THE ORIGINAL MAC FORMULATION.
      LOGICAL TA,TB,TC,TV,TH,TEST
      LOGICAL ANT,AA,AB,AC,AD
      CODE ANALYSIS
64    DO 20 L=2,LMAXZ
      KMM=L*KMAX
      DO 20 K=2,KMAXZ
      KT=K+KMM
      KM=K.T-KMAX
      KB=KM-KMAX
      IF(SUR(KM))21,22
21    IF(NPRTA(KM))20,24,20
24    CONTINUE
      EMP(KM)=1
      ARB(KM)=0
      SUR(KM)=0
      P(KM)=0.0
      IF(EMP(KM+1))25,26
25    CONTINUE
      U(KM)=0.0
26    IF(EMP(KM-1))27,28
27    CONTINUE
      U(KM-1)=0.0
28    IF(EMP(KT))29,30
29    CONTINUE
      V(KM)=0.0
30    IF(EMP(KB))31,20
31    CONTINUE
      V(KB)=0.0
      GO TO 20
22    IF(EMP(KM))32,20
32    CONTINUE
      P(KM)=0.0
      NNN=NPRTA(KM)
      IF(NNN)33,20,33
33    CONTINUE
      SUR(KM)=1
      EMP(KM)=0
      J=KM
      VBAR=SLUA(J)
      VBAR=VBAR/NNN
      UBAR=SUUA(J)
      UBAR=UBAR/NNN
      IF(EMP(KM+1).OR.OUT(KM+1))34,35
34    CONTINUE
      U(KM)=UBAR
35    IF(EMP(KM-1).OR.OUT(KM-1))36,37
36    CONTINUE
      U(KM-1)=UBAR
37    IF(EMP(KT).OR.OUT(KT))38,39
38    CONTINUE
      V(KM)=VBAR
39    IF(EMP(KB).OR.OUT(KB))40,20

```

```

40 CONTINUE
V(KB)=VBAR
20 CONTINUE
DO 41 L=2,LMAXZ
KMM=L*KMAX
DO 41 K=2,KMAXZ
KT=K+KMM
KM=KT-KMAX
KB=KM-KMAX
IF(NPRTA(KM))42,41,42
42 IF(FULL(KM))43,44
43 IF(EMP(KM+1))45,46
46 IF(EMP(KM-1))45,47
47 IF(EMP(KT))45,48
48 IF(EMP(KB))45,41
45 CONTINUE
SUR(KM)=1
FULL(KM)=0
P(KM)=0.0
GO TO 41
44 CONTINUE
IF(OB(KM))41,87
87 IF(SUR(KM))50,41
50 SGP=0.0
P(KM)=0.
N=N+1
IF(EMP(KM+1))41,51
51 IF(FULL(KM+1))52,53
52 SGP=SGP+P(KM+1)
N=N+1
53 IF(EMP(KM-1))41,54
54 IF(FULL(KM-1))55,56
55 SGP=SGP+P(KM-1)
N=N+1
56 IF(EMP(KT))41,57
57 IF(FULL(KT))58,59
58 SGP=SGP+P(KT)
N=N+1
59 IF(EMP(KB))41,60
60 IF(FULL(KB))61,62
61 SGP=SGP+P(KB)
N=N+1
62 FULL(KM)=1
GAS(KM)=0
SUR(KM)=0
IF(N)66,66,67
66 P(KM)=0.
GO TO 41
67 P(KM)=SGP/N
41 CONTINUE
C
C
C
DETERMINE WHEN PRESSURE COMPUTATION IS NECESSARY IN ARBITRARY
BOUNDARY CELLS 1) FLUID IS IN CONTACT IE ARB=1 2) CELL DOES NOT
CONTAIN THE INTERSECTION OF A FREE SURFACE AND THE BOUNDARY.
DO 90 L=2,LMAXZ
KTT=L*KMAX
DO 91 K=2,KMAXZ
KT=KTT+K
KM=KT-KMAX
KB=KM-KMAX

```

```

126 IF(OB(KM))126,91
AA=OK(KM)
TA=OB(KM+1).INT.EMP(KM+1)
TB=OB(KT).INT.EMP(KT)
TC=OB(KM-1).INT.EMP(KM-1)
TV=OB(KB).INT.EMP(KB)
TEST=TA.UN.TB.UN.TC.UN.TV
OK(KM)=OB(KM).INT.ARB(KM).INT.(.NOT.TEST)
TEST=0
IF(DNORX(KM))120,121,122
120 TEST=EMP(KM-1)
GO TO 121
122 TEST=EMP(KM+1)
121 IF(DNORY(KM))123,124,125
123 TEST=TEST.UN.EMP(KB)
GO TO 124
125 TEST=TEST.UN.EMP(KT)
124 CONTINUE
OK(KM)=OK(KM).INT.(.NOT.TEST)
IF(OK(KM))103,104
104 P(KM)=0.
GO TO 91
C
C THE FIRST TIME CELLS PRESSURE IS TO BE COMPUTED LOAD CELL
C PRESSURE WITH AVERAGE OF ITS NEIGHBORS.
103 TEST=(AA.EQ.0).INT.OK(KM)
IF(TEST)92,91
92 N=0
SP=0.
IF(OK(KT).UN.FULL(KT))93,94
93 SP=SP+P(KT)
N=N+1
94 IF(OK(KB).UN.FULL(KB))95,96
95 SP=SP+P(KB)
N=N+1
96 IF(OK(KM+1).UN.FULL(KM+1))97,98
97 SP=SP+P(KM+1)
N=N+1
98 IF(OK(KM-1).UN.FULL(KM-1))99,101
99 SP=SP+P(KM-1)
N=N+1
101 IF(N)91,91,102
102 P(KM)=SP/N
91 CONTINUE
90 CONTINUE
CALL VEL55
DO 73 L=2,LMAXZ
KMM=(L-1)*KMAX
DO 73 K=2,KMAXZ
KM=K+KMM
IF(EMP(KM).OR.SUR(KM))76,73
76 IF(ARB(KM))73,88
88 P(KM)=PAMB
73 CONTINUE
RETURN
END

```

```

*      LIST8
*      FORTRAN          VEL
      SUBROUTINE VEL (NSIG)
      USE CMAC
      CODE ANALYSIS

C      THIS ROUTINE COMPUTES ALL OF THE CELL VELOCITIES.  THE
C      PRESSURE GRADIENTS ARE CALCULATED FOR BOUNDARY PRESSURES
C      LOCATED AT THE MID POINTS OF THE LAGRANGE BOUNDARY SEGMENTS.
C      APPROXIMATE LINEAR INTERPOLATION USED.  THIS ROUTINE AND VELD ARE
C      THE SAME EXCEPT VELD COMPUTES ONLY A SINGLE CELL.

      LOGICAL TEST
      DO 192 L=2,LMAXZ
      KTT=L*KMAX
      DO 192 K=2,KMAXZ
      KT=K+KTT
      KM=KT-KMAX
      KB=KM-KMAX
      KTZ=KT+KMAX
      TEST=FULL(KM).UN.(SUR(KM).INT.(.NOT.COR(KM)))
      IF(TEST)20,192

C      U COMP.
20  IF(BND(KM+1))25,36
36  IF(EMP(KM+1).UN.COR(KM+1))37,21
37  GO TO(25,42),NSIG
42  U(KM)=U(KM)-GR*DT
      GO TO 25
21  CONTINUE
      XC=R(K)
      IF(OK(KM).INT.COR(KM-1))70,71
70  XC=PNORX(KM)
71  CONTINUE
      XA=R(K+1)
      IF(OK(KM+1).INT.COR(KM+2))72,73
72  XA=PNORX(KM+1)
73  CONTINUE
      PA=P(KM)
      IF(OK(KM).INT.COR(KT))88,89
88  P2=P(KM)
      P1=P(KB)
      Y2=PNORY(KM)
      Y1=Z(L-1)
      GO TO 90
89  IF(OK(KM).INT.COR(KB))91,92
91  P2=P(KT)
      P1=P(KM)
      Y2=Z(L+1)
      Y1=PNORY(KM)
90  SLP=(P2-P1)/(Y2-Y1)
      ORD=P1-SLP*Y1
      XC=R(K)
      PA=SLP*Z(L)+ORD
92  CONTINUE
      PB=P(KM+1)
      IF(OK(KM+1).INT.COR(KT+1))93,94
93  P2=P(KM+1)
      P1=P(KB+1)
      Y2=PNORY(KM+1)

```



```

      Y1=Z(L-1)
      GO TO 95
94  IF(OK(KM+1)).INT.COR(KB+1))96,97
96  P2=P(KT+1)
      P1=P(KM+1)
      Y2=Z(L+1)
      Y1=PNORY(KM+1)
95  SLP=(P2-P1)/(Y2-Y1)
      ORD=P1-SLP*Y1
      XA=R(K+1)
      PD=GLP*Z(L)+ORD
97  CONTINUE
      U(KM)=ETA(KM)+DT*(PA-PB)/(XA-XC)
      U(KM)=U(KM)/RHOA-GR*DT
C
C  V COMP.
25  IF(BND(KT))192,24
24  IF(EMP(KT)).UN.COR(KT))22,23
22  GO TO(192,45),NSIG
45  V(KM)=V(KM)-GZ*DT
      GO TO 192
23  CONTINUE
      YC=Z(L)
      IF(OK(KM)).INT.COR(KB))74,75
74  YC=PNORY(KM)
75  CONTINUE
      YA=Z(L+1)
      IF(OK(KT)).INT.COR(KTZ))76,77
76  YA=PNORY(KT)
77  CONTINUE
      PA=P(KM)
      IF(OK(KM)).INT.COR(KM+1))78,79
78  P2=P(KM)
      P1=P(KM-1)
      X2=PNORX(KM)
      X1=R(K-1)
      GO TO 80
79  IF(OK(KM)).INT.COR(KM-1))81,82
81  P2=P(KM+1)
      P1=P(KM)
      X1=PNORX(KM)
      X2=R(K+1)
80  SLP=(P2-P1)/(X2-X1)
      ORD=P1-SLP*X1
      YC=Z(L)
      PA=SLP*R(K)+ORD
82  CONTINUE
      PB=P(KT)
      IF(OK(KT)).INT.COR(KT+1))83,84
83  P2=P(KT)
      P1=P(KT-1)
      X2=PNORX(KT)
      X1=R(K-1)
      GO TO 85
84  IF(OK(KT)).INT.COR(KT-1))86,87
86  P2=P(KT+1)
      P1=P(KT)
      X2=R(K+1)
      X1=PNORX(KT)
85  SLP=(P2-P1)/(X2-X1)

```

```

ORD=P1-SLP*X1
YA=Z(L+1)
PB=SLP*R(K)+ORD
87 CONTINUE
V(KM)=PSI(KM)+DT*(PA-PB)/(YA-YC)
V(KM)=V(KM)/RHOA-GZ*DT
192 CONTINUE
C
C C
C C C
USE STANDARD MAC TREATMENT OF FREE SURFACE CELLS.
DO 26 L=2,LMAXZ
KMM=L*KMAX
DO 26 K=2,KMAXZ
KT=K+KMM
KM=KT-KMAX
KE=KM-KMAX
IF(SUR(KM).OR.OK(KM))27,26
27 IF(EMP(KM-1).OR.COR(KM-1))28,29
28 U(KM-1)=U(KM)*(RB(K)/RB(K-1))**NEXP
GO TO 31
29 IF(EMP(KM+1).OR.COR(KM+1))32,31
32 U(KM)=U(KM-1)*(RB(K-1)/RB(K))**NEXP
31 IF(EMP(KB).OR.COR(KB))33,34
33 V(KB)=V(KM)
GO TO 26
34 IF(EMP(KT).OR.COR(KT))35,26
35 V(KM)=V(KB)
26 CONTINUE
CALL VEL55
C
BOUNDARY CONDITIONS ON VELOCITIES AT EDGE OF MESH
DO 193 K=2,KMAXZ
KT=K+(LMAXZ-1)*KMAX
KTB=KT-KMAX
KTT=KT+KMAX
KB=K
KBT=KB+KMAX
IF(IN(KTT))50,51
50 V(KT)=VIN
V(KTT)=V(KT)
U(KTT)=0.
GO TO 52
51 IF(OUT(KTT))53,54
53 V(KT)=V(KTB)
U(KTT)=U(KT)
V(KTT)=V(KTB)
GO TO 52
54 IF(BND(KTT))700,52
700 V(KT)=0.
U(KTT)=U(KT)
V(KTT)=-V(KTB)
52 IF(IN(KB))55,56
55 V(KB)=VIN
U(KB)=0.
GO TO 193
56 IF(OUT(KB))58,59
58 V(KB)=V(KBT)
U(KB)=U(KBT)
GO TO 193
59 IF(BND(KB))710,193
710 V(KB)=0.

```

```

      U(KB)=U(KBT)
193  CONTINUE
      DO 194 L=2,LMAXZ
      KK=(L-1)*KMAX
      LR=KK+KMAXZ
      LL=2+KK
      IF(IN(LL-1))60,61
60   U(LL-1)=UIN
      V(LL-1)=0.
      GO TO 62
61   IF(OUT(LL-1))63,64
63   U(LL-1)=U(LL)
      V(LL-1)=V(LL)
      GO TO 62
64   IF(BND(LL-1))720,62
720  U(LL-1)=0.
      V(LL-1)=V(LL)
62   IF(IN(LR+1))65,66
65   U(LR)=UIN
      U(LR+1)=U(LR)
      V(LR+1)=V(LR)
      GO TO 194
66   IF(OUT(LR+1))67,68
67   U(LR)=U(LR-1)
      U(LR+1)=U(LR)
      V(LR+1)=V(LR)
      GO TO 194
68   IF(BND(LR+1))730,194
730  U(LR)=0.
      U(LR+1)=-U(LR-1)
      V(LR+1)=V(LR)
194  CONTINUE
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          VELP
SUBROUTINE VELP(NSIG,KM,K,L)
  USE CMAC
  CODE ANALYSIS
C      THIS ROUTINE COMPUTES U AND V FOR A SINGLE CELL
C      THE PRESSURE GRADIENTS ARE CALCULATED FOR BOUNDARY PRESSURES
C      LOCATED AT THE MID POINTS OF THE LAGRANGE BOUNDARY SEGMENTS.
C      APPROXIMATE LINEAR INTERPOLATION USED.
      VMAX=0.0
      KT=KM+KMAX
      KB=KM-KMAX
      KTZ=KT+KMAX
      LOGICAL TEST
      TEST=FULL(KM).UN.(SUR(KM).INT.(.NOT.COR(KM)))
      IF(TEST)20,192
C
C      U COMP.
20  IF(BND(KM+1))25,36
36  IF(EMP(KM+1).UN.COR(KM+1))37,21
37  GO TO(25,42),NSIG
42  U(KM)=U(KM)-GR*DT
      GO TO 25
21  CONTINUE
      XC=R(K)
      IF(OK(KM).INT.COR(KM-1))70,71
70  XC=PNORX(KM)
71  CONTINUE
      XA=R(K+1)
      IF(OK(KM+1).INT.COR(KM+2))72,73
72  XA=PNORX(KM+1)
73  CONTINUE
      PA=P(KM)
      IF(OK(KM).INT.COR(KT))88,89
88  P2=P(KM)
      P1=P(KB)
      Y2=PNORY(KM)
      Y1=Z(L-1)
      GO TO 90
89  IF(OK(KM).INT.COR(KB))91,92
91  P2=P(KT)
      P1=P(KM)
      Y2=Z(L+1)
      Y1=PNORY(KM)
90  SLP=(P2-P1)/(Y2-Y1)
      ORD=P1-SLP*Y1
      XC=R(K)
      PA=SLP*Z(L)+ORD
92  CONTINUE
      PB=P(KM+1)
      IF(OK(KM+1).INT.COR(KT+1))93,94
93  P2=P(KM+1)
      P1=P(KB+1)
      Y2=PNORY(KM+1)
      Y1=Z(L-1)
      GO TO 95
94  IF(OK(KM+1).INT.COR(KB+1))96,97
96  P2=P(KT+1)

```

```

P1=P(KM+1)
Y2=Z(L+1)
Y1=PNORY(KM+1)
95 SLP=(P2-P1)/(Y2-Y1)
ORD=P1-SLP*Y1
XA=R(K+1)
PB=SLP*Z(L)+ORD
97 CONTINUE
U(KM)=ETA(KM)+DT*(PA-PB)/(XA-XC)
U(KM)=U(KM)/RHOA-GR*DT
C
C
V COMP
25 IF(BND(KT))192,24
24 IF(EMP(KT).UN.COR(KT))22,23
22 GO TO(192,45),NSIG
45 V(KM)=V(KM)-GZ*DT
GO TO 192
23 CONTINUE
YC=Z(L)
IF(OK(KM).INT.COR(KB))74,75
74 YC=PNORY(KM)
75 CONTINUE
YA=Z(L+1)
IF(OK(KT).INT.COR(KTZ))76,77
76 YA=PNORY(KT)
77 CONTINUE
PA=P(KM)
IF(OK(KM).INT.COR(KM+1))78,79
78 P2=P(KM)
P1=P(KM-1)
X2=PNORX(KM)
X1=R(K-1)
GO TO 80
79 IF(OK(KM).INT.COR(KM-1))81,82
81 P2=P(KM+1)
P1=P(KM)
X1=PNORX(KM)
X2=R(K+1)
80 SLP=(P2-P1)/(X2-X1)
ORD=P1-SLP*X1
YC=Z(L)
PA=SLP*R(K)+ORD
82 CONTINUE
PB=P(KT)
IF(OK(KT).INT.COR(KT+1))83,84
83 P2=P(KT)
P1=P(KT-1)
X2=PNORX(KT)
X1=R(K-1)
GO TO 85
84 IF(OK(KT).INT.COR(KT-1))86,87
86 P2=P(KT+1)
P1=P(KT)
X2=R(K+1)
X1=PNORX(KT)
85 SLP=(P2-P1)/(X2-X1)
ORD=P1-SLP*X1
YA=Z(L+1)
PB=SLP*R(K)+ORD
87 CONTINUE

```

```

V(KM)=PSI(KM)+DT*(PA-PB)/(YA-YC)
V(KM)=V(KM)/RHOA-GZ*DT
UMAX=ABSF(U(KM))+ABSF(V(KM))
VMAX=.5*(ABSF(VMAX-UMAX)+VMAX+UMAX)
192 CONTINUE
C
C USE STANDARD MAC TREATMENT OF FREE SURFACE CELL VELOCITIES.
IF(SUR(KM).OR.OK(KM))27,26
27 IF(EMP(KM-1).OR.COR(KM-1))28,29
28 U(KM-1)=U(KM)*(RB(K)/RB(K-1))**NEXP
GO TO 31
29 IF(EMP(KM+1).OR.COR(KM+1))32,31
32 U(KM)=U(KM-1)*(RB(K-1)/RB(K))**NEXP
31 IF(EMP(KB).OR.COR(KB))33,34
33 V(KB)=V(KM)
GO TO 26
34 IF(EMP(KT).OR.COR(KT))35,26
35 V(KM)=V(KB)
26 CONTINUE
CALL VEL55P(KM,K,L)
RETURN
END

```

```

*      LIST8
*      FORTRAN          VEL55
      SUBROUTINE VEL55
      USE CMAC
      CODE ANALYSIS

C      THIS ROUTINE COMPUTES VELOCITY FOR SURFACE CELLS WITH ONE SIDE
C      OPEN.  ADJUSTS TO MAKE DIVERGENCE VANISH.
C

      LOGICAL TEST
      NM 1 1=2,1MAX7
      KMM=L*KMAX
      DO 1 K=2,KMAXZ
      KT=K+KMM
      KM=KT-KMAX
      KB=KM-KMAX
      IF(SUR(KM).OR.OK(KM))2,1
2     TEST=(.NOT.(EMP(KM+1).UN.EMP(KT).UN.EMP(KB))) .INT.
1     (EMP(KM-1).UN.COR(KM-1)) .INT. (.NOT.BND(KM-1)) .INT.
1     (.NOT.(COR(KM+1).UN.COR(KT).UN.COR(KB)))
      IF(TEST)3,4
3     U(KM-1)=R(K)**NEXP*DR(K)*(V(KM)-V(KB))/(RB(K-1)**NEXP*DZ(L))
      U(KM-1)=U(KM-1)+U(KM)*(RB(K)/RB(K-1))**NEXP
      GO TO 1
4     TEST=(.NOT.(EMP(KM-1).UN.EMP(KT).UN.EMP(KB))) .INT.
1     (EMP(KM+1).UN.COR(KM+1)) .INT. (.NOT.BND(KM+1)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KT).UN.COR(KB)))
      IF(TEST)5,6
5     U(KM)=U(KM-1)*(RB(K-1)/RB(K))**NEXP
      U(KM)=U(KM)-R(K)**NEXP*DR(K)*(V(KM)-V(KB))/(RB(K)**NEXP*DZ(L))
      GO TO 1
6     TEST=(.NOT.(EMP(KM-1).UN.EMP(KM+1).UN.EMP(KT))) .INT.
1     (EMP(KB).UN.COR(KB)) .INT. (.NOT.BND(KB)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KM+1).UN.COR(KT)))
      IF(TEST)7,8
7     V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
1     /(R(K)**NEXP*DR(K))
      GO TO 1
8     TEST=(.NOT.(EMP(KM-1).UN.EMP(KM+1).UN.EMP(KB))) .INT.
1     (EMP(KT).UN.COR(KT)) .INT. (.NOT.BND(KT)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KM+1).UN.COR(KB)))
      IF(TEST)9,10
9     V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
1     /(R(K)**NEXP*DR(K))
      GO TO 1
10    CONTINUE
1     CONTINUE
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          VEL55P
SUBROUTINE VEL55P(KM,K,L)
USE CMAC
CODE ANALYSIS

C      THIS ROUTINE COMPUTES VELOCITY FOR SURFACE CELLS WITH ONE SIDE
C      OPEN.  ADJUSTS TO MAKE DIVERGENCE VANISH.
C      THIS ROUTINE IS THE SAME AS VELL55 EXCEPT THAT IT COMPUTES
C      ONLY THE CELL SPECIFIED IN THE CALLING ARGUMENT.

      LOGICAL TEST
      KT=KM+KMAX
      KB=KM-KMAX
      IF(SUR(KM).OR.OK(KM))2,1
2     TEST=(.NOT.(EMP(KM+1).UN.EMP(KT).UN.EMP(KB))) .INT.
1     (EMP(KM-1).UN.COR(KM-1)) .INT. (.NOT.BND(KM-1)) .INT.
1     (.NOT.(COR(KM+1).UN.COR(KT).UN.COR(KB)))
      IF(TEST)3,4
3     U(KM-1)=R(K)**NEXP*DR(K)*(V(KM)-V(KB))/(RB(K-1)**NEXP*DZ(L))
      U(KM-1)=U(KM-1)+U(KM)*(RB(K)/RB(K-1))**NEXP
      GO TO 1
4     TEST=(.NOT.(EMP(KM-1).UN.EMP(KT).UN.EMP(KB))) .INT.
1     (EMP(KM+1).UN.COR(KM+1)) .INT. (.NOT.BND(KM+1)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KT).UN.COR(KB)))
      IF(TEST)5,6
5     U(KM)=U(KM-1)*(RB(K-1)/RB(K))**NEXP
      U(KM)=U(KM)-R(K)**NEXP*DR(K)*(V(KM)-V(KB))/(RB(K)**NEXP*DZ(L))
      GO TO 1
6     TEST=(.NOT.(EMP(KM-1).UN.EMP(KM+1).UN.EMP(KT))) .INT.
1     (EMP(KB).UN.COR(KB)) .INT. (.NOT.BND(KB)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KM+1).UN.COR(KT)))
      IF(TEST)7,8
7     V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
1     /(R(K)**NEXP*DR(K))
      GO TO 1
8     TEST=(.NOT.(EMP(KM-1).UN.EMP(KM+1).UN.EMP(KB))) .INT.
1     (EMP(KT).UN.COR(KT)) .INT. (.NOT.BND(KT)) .INT.
1     (.NOT.(COR(KM-1).UN.COR(KM+1).UN.COR(KB)))
      IF(TEST)9,10
9     V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
1     /(R(K)**NEXP*DR(K))
      GO TO 1
10    CONTINUE
1     CONTINUE
      RETURN
      END

```



```

*      LIST8
*      FORTRAN          VEL56
SUBROUTINE VEL56 (XR,XZ,UP,VP)
USE CMAC
CODE ANALYSIS

C
C
C      THIS ROUTINE USES THE MAC WEIGHTING FORMULA TO COMPUTE THE
C      VELOCITY AT ANY POINT - SAME AS THAT USED IN ROUTINES TRAJ AND
C      VELD AND VEL.

      IB=3
      IA=3
      LP=XZ/DZ(1)+2
      KP=XR/DR(1)+2
      NT= 1 P*KMAX+KP
      NM=NT-KMAX
      NB=NM-KMAX
      DLLR=RB(KP)-RB(KP-1)
      DLLZ=ZB(LP)-ZB(LP-1)
      IF( XZ -Z(LP))52,52,53
52 LB=LP-1
   GO TO 69
53 CONTINUE
   LB=LP
69 KT= LB*KMAX+KP
   KM= (LB-1)*KMAX+KP
   SX=(R(KP)- XR )/DLLR
   SY=(ZB(LB)- XZ )/DLLZ
   IF(U(KT-1))54,55,54
55 U1=U(NM-1)
   GO TO 56
54 U1=U(KT-1)
56 IF(U(KT))57,58,57
58 U2=U(NM)
   GO TO 59
57 U2=U(KT)
59 IF(U(KM-1))60,61,60
61 U3=U(NM-1)
   GO TO 62
60 U3=U(KM-1)
62 IF(U(KM))63,64,63
64 U4=U(NM)
   GO TO 65
63 U4=U(KM)
65 UP=(.5+SX)*( .5-SY)*U1 + (.5-SX)*( .5-SY)*U2
   UP=UP + (.5+SX)*( .5+SY)*U3 + (.5-SX)*( .5+SY)*U4
   IF( XR -R(KP))66,66,67
66 KB=KP-1
   GO TO 68
67 KB=KP
68 KD=(LP-2)*KMAX+KB
   KM=(LP-1)*KMAX+KB
   SX=(RB(KB)- XR )/DLLR
   SY=(Z(LP)- XZ )/DLLZ
   IF(V(KM))70,71,70
71 V1=V(NM)
   GO TO 72
70 V1=V(KM)
72 IF(V(KM+1))73,74,73

```

```

74 V2=V(NM)
   GO TO 75
73 V2=V(KM+1)
75 IF(V(KD))77,76,77
76 V3=V(NB)
   GO TO 78
77 V3=V(KD)
78 IF(V(KD+1))79,80,79
80 V4=V(NB)
   GO TO 81
79 V4=V(KD+1)
81 VP= (.5+SX)*(.5-SY)*V1 + (.5-SX)*(.5-SY)*V2
   VP= VP + (.5+SX)*(.5+SY)*V3 + (.5-SX)*(.5+SY)*V4
   RETURN
   END

```

```

*      LIST8
*      FORTRAN          VEL57
      SUBROUTINE VEL57
      USE CMAC
      USE PACK
      CODE ANALYSIS

      THIS ROUTINE TAKES CARE OF CELLS IN WHICH A FREE SURFACE AND
      AN ARBITRARY BOUNDARY INTERSECT. PROVISION ADDED FOR LATER
      ADDITION OF MOVING BOUNDARY, THEREFORE MAKE CERTAIN UBND
      AND VBND ARE ZERO

      LOGICAL TEST,TA
      DO 20 J=1,NGRID
      SNC(J)=SUR(J).INT.(.NOT.COR(J))
20    CONTINUE
      DO 92 L=2,LMAXZ
      KTT=L*KMAX
      DO 92 K=2,KMAXZ
      KT=KTT+K
      KM=KT-KMAX
      KB=KM-KMAX

      C      DETECT INTERSECTION.
      C      TEST=OB(KM).INT.ARB(KM).INT.(.NOT.OK(KM)).INT.(.NOT.EMP(KM))
      C      IF(TEST)93,92

      C      CELL CONTAINS INTERSECTION. DETECT CASE WHERE FLUID IS
      C      SEPARATING FROM THE BOUNDARY.
93    CONTINUE
      PN=PNORX(KM)
      PNY=PNORY(KM)
      DNX=DNORX(KM)
      DNY=DNORY(KM)
      CALL VEL56(PN,PNY,UP,VP)
      UP=UP-UBND
      VP=VP-VBND
      DOT=UP*DNX+VP*DNY
      IF(DOT)94,92,92

      C      FLUID MOVING TOWARDS BOUNDARY. DETECT CASES WHERE TWO OF THE
      C      CELLS SIDES ARE IN CONTACT WITH FLUID. ADJUST OTHER TWO
      C      VELOCITIES TO BRING VELOCITY AT MID POINT OF BOUNDARY SEGMENT
      C      TANGENT TO SEGMENT. MAINTAIN VANISHING DIVERGENCE IN CELL.
94    TEST=(FULL(KM+1).UN.SNC(KM+1)).INT.(FULL(KT).UN.SNC(KT))
      TEST=TEST.INT.(EMP(KM-1).UN.COR(KM-1)).INT.(EMP(KB).UN.COR(KB))
      TA=(FULL(KM+1).UN.SNC(KM+1)).INT.(FULL(KB).UN.SNC(KB))
      TA=TA.INT.(EMP(KM-1).UN.COR(KM-1)).INT.(EMP(KT).UN.COR(KT))
      TEST=TEST.UN.TA
      IF(TEST)103,104
103   UA=U(KM-1)
      DU=.1*VMAX+.0001
      U(KM-1)=UA+DU
      IF(EMP(KB).UN.COR(KB))116,117
116   V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
      1/(R(K)**NEXP*DR(K))
      GO TO 118
117   V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
      1/(R(K)**NEXP*DR(K))

```

```

118 CALL VEL56(PNX,PNY,UP,VP)
    UP=UP-UBND
    VP=VP-VBND
    DOTP=UP*DNX+VP*DNY
    DRIV=(DOTP-DOT)/DU
    IF(DRIV)105,106,105
105 U(KM-1)=.5*(UA+U(KM-1))-.5*(DOT+DOTP)/DRIV
    IF(EMP(KB).UN.COR(KB))107,108
107 V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
    GO TO 92
108 V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
    GO TO 92
104 TEST=(FULL(KM-1).UN.SNC(KM-1)).INT.(FULL(KT).UN.SNC(KT))
    TEST=TEST.INT.(EMP(KM+1).UN.COR(KM+1)).INT.(EMP(KB).UN.COR(KB))
    TA=(FULL(KM-1).UN.SNC(KM-1)).INT.(FULL(KB).UN.SNC(KB))
    TA=TA.INT.(EMP(KM+1).UN.COR(KM+1)).INT.(EMP(KT).UN.COR(KT))
    TEST=TEST.UN.TA
    IF(TEST)109,100
109 UA=U(KM)
    DU=.1*VMAX+.0001
    U(KM)=UA+DU
    IF(EMP(KB).UN.COR(KB))126,127
126 V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
    GO TO 128
127 V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
128 CALL VEL56(PNX,PNY,UP,VP)
    UP=UP-UBND
    VP=VP-VBND
    DOTP=UP*DNX+VP*DNY
    DRIV=(DOTP-DOT)/DU
    IF(DRIV)110,106,110
110 U(KM)=.5*(UA+U(KM))-.5*(DOT+DOTP)/DRIV
    IF(EMP(KB).UN.COR(KB))111,112
111 V(KB)=V(KM)+DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
    GO TO 92
112 V(KM)=V(KB)-DZ(L)*(RB(K)**NEXP*U(KM)-RB(K-1)**NEXP*U(KM-1))
    1/(R(K)**NEXP*DR(K))
    GO TO 92
106 WOT 3,114,K,L,NCYCLE
114 FORMAT(5X19HDRIV IN REFLAG ZERO,318)
    GO TO 92

```

C
C
C

```

    ONLY ONE SIDE OR LESS IN CONTACT WITH FLUID.  DETECT ONE SIDE
    CASE.
100 TEST=FULL(KM+1).UN.SNC(KM+1)
    TEST=TEST.INT.(FULL(KM-1).UN.SNC(KM-1))
    IF(TEST)129,130
129 VA=V(KM)
    DU=.1*VMAX+.0001
    V(KM)=VA+DU
    V(KB)=VA+DU
    CALL VEL56(PNX,PNY,UP,VP)
    UP=UP-UBND
    VP=VP-VBND
    DOTP=UP*DNX+VP*DNY.

```

```

      DRIV=(DOTP-DOT)/DU
      IF(DRIV)131,106,131
131  V(KM)=.5*(VA+V(KM))-.5*(DOTP+DOT)/DRIV
      V(KB)=V(KM)
      GO TO 92
130  TEST=FULL(KT).UN.SNC(KT)
      TEST=TEST.INT.(FULL(KB).UN.SNC(KB))
      IF(TEST)132,134
132  UA=U(KM)
      DU=.1*VMAX+.0001
      U(KM)=UA+DU
      U(KM-1)=UA+DU
      CALL VEL56(PNX,PNY,UP,VP)
      UP=UP-UBND
      VP=VP-VBND
      DOTP=UP*DNX+VP*DNY
      DRIV=(DOTP-DOT)/DU
      IF(DRIV)133,106,133
133  U(KM)=.5*(UA+U(KM))-.5*(DOTP+DOT)/DRIV
      U(KM-1)=U(KM)
      GO TO 92
C
C 134  ISOLATED PARTICLE STRIKING BOUNDARY.
      NNN=NPRTA(KM)+NPRTB(KM)
      VP=SLUA(KM)
      VP=VP/NNN
      UP=SLUA(KM)
      UP=UP/NNN
      UP=UP-UBND
      VP=VP-VBND
      DOT=UP*DNORX(KM)+VP*DNORY(KM)
      IF(DOT)115,92,92
115  VTAN=UP*DNORY(KM)-VP*DNORX(KM)-(GR*DNORY(KM)-GZ*DNORX(KM))*DT
      U(KM)=VTAN*DNORY(KM)+UBND
      U(KM-1)=U(KM)
      V(KM)=-VTAN*DNORX(KM)+VBND
      V(KB)=V(KM)
92  CONTINUE
      RETURN
      END

```

```

*      LIST8
*      FORTRAN          SAM
      SUBROUTINE SAM
      GENERIC Q8QFLA
      USE CMAC
      CODE ANALYSIS
C
C      THIS ROUTINE SAVES COMMON ON TAPE
C
      IB=5
      WORD=0
      IF(Q8QFLA(1))36,37
37      CONTINUE
      IF (NTPE) 20,21,22
28      CONTINUE
      PRINT 31, NTAPE
      NTPE = -1
      CALL UNLOAD(IB)
20      CONTINUE
      CALL ASSIGN(IB,1,NHE)
      NTAPE = NHE
      GO TO 12
21      CONTINUE
      CALL ASSIGN(IB,1,NHD)
      NTAPE = NHD
      GO TO 12
36      CONTINUE
      PRINT 38
38      FORMAT (20HSENSE SWITCH ONE SET )
22      CONTINUE
      CALL ASSIGN(IB,1,NHC)
      NTAPE = NHC
12      CONTINUE
11      BUFFER OUT(IB,1)(NVAC,FLAG(KYA))
2      IF(UNIT,IB,WORD)2,3,7,7
7      CONTINUE
      IF (NTPE) 23,24,25
23      CONTINUE
      PRINT 31, NTAPE
      PRINT 32
31      FORMAT (20HBAD OUT FILE TAPE , A10)
32      FORMAT (20HEXIT SECOND BAD TAPE )
      CALL EXIT
24      CONTINUE
      IF (NTP1) 23,23,26
26      CONTINUE
      NTP2 = -1
      NTP3 = NTP1
      NTP1 = 0
      GO TO 28
25      CONTINUE
      IF (NTP2) 23,27,23
27      CONTINUE
      NTP1 = -1
      NTP3 = NTP2
      NTP2 = 1
      GO TO 28
3      CONTINUE
      BUFFER OUT(IB,1)(RP(1),PNORY(KYA))
40      IF(UNIT,IB,WORD)40,41,7,7

```

```

41 CONTINUE
   IF (NTPE) 33,34,35
33 CONTINUE
   NTPE = NTP3
   GO TO 29
34 CONTINUE
   NTPE = NTP1
   GO TO 29
35 CONTINUE
   NTPE = NTP2
29 CONTINUE
   CALL UNLOAD(1B)
   PRINT 10, NCYCLE, TIME, NTAPE
10 FORMAT (14H DUMP AT CYCLE 16,4HTIME E16.8,8H TAPE,A10)
   RETURN
   END

```

```

*      LIST8
*      FORTRAN          RDX
      SUBROUTINE RDX
      CODE ANALYSIS
C
C      THIS ROUTINE READS COMMON FROM PREVIOUS DUMP TAPE
C
      WORD=0
      USE CMAC
      NTP1 = 1
      NTP2 = 0
      IB=5
11  FORMAT (3R5)
5   FORMAT(5HMACH )
      PRINT 5
      RIT 2,11, NHC, NHD, NHE
      NTAPE = NHC
      CALL ASSIGN (IB,1,NHC)
      NTPE = 0
32  BUFFER IN(IB,1)(NVAC,FLAG(KYA))
34  IF(UNIT,IB,WORD)34,35,35,37
37  PRINT 41, NTAPE
41  FORMAT (20HBAD INFILE      TAPE      ,A10)
      CALL UNLOAD(IB)
      IF (NTPE) 42,43,42
43  CONTINUE
      CALL ASSIGN(IB,1,NHD)
      NTAPE = NHD
      NTPE = 1
      MM=0
      GO TO 32
42  CONTINUE
      PRINT 45
45  FORMAT (20HEXIT SECOND BAD READ )
      CALL EXIT
35  CONTINUE
      BUFFER IN(IB,1)(RP(1),PNORY(KYA))
50  IF(UNIT,IB,WORD)50,51,51,37
51  CONTINUE
      PRINT 10,NCYCLE TIME,NTAPE
10  FORMAT (14HSTART AT CYCLE      16,4HTIME      E16.8,8H      TAPE,A10)
      CALL UNLOAD(IB)
      RETURN
      END

```



```

*      LIST8
*      FORTRAN          EDIT
      SUBROUTINE EDIT
      CODE ANALYSIS
      USE CMAC
      USE PACK
      IB=3

C
      WOT 59,35 $ CALL 000TBF
36  FORMAT(4HEDIT)
C
460  FORMAT(3H L=,15,5X,39HK,ETA,PSI,U,V,P,X OR R, UPART,VPART )
461  FORMAT(14,8E14.6)
462  FORMAT(6H CYCLE,18,5H TIME,E15.6,3H DT,E15.6,6H NPART,110,6H PAMB
1,E15.7)
463  FORMAT(3H L=,15,5X,87HK,FULL,SUR,EMP,OB,OK,ARB,COR,BND,IN,OUT,GAS,
1EMPBND,SNC,NPT,DNORX,DNORY,PNORX,PNORY,KMPR)
465  FORMAT(15,15,212,15,312,15,212,15,212,15,212,2X,15,4E14.6,17)
410  WOT 1B,462,NCYCLE,TIME,DT,NPART,PAMB
      WOT 1B,11,PGASZ,GASVZ,GAMA,PGAS,GASV,ITEST
11  FORMAT(5X6HPGASZ,E15.7,7H GASVZ,E15.7,6H GAMA,E15.7,6H PGAS,
1E15.7,6H GASV,E15.7,4H IT,11)
      WOT 1B,467,RHOA,MU
467  FORMAT(5X4HRHO=,E14.6,3H MU=,E14.6)
      WOT 1B,70
70  FORMAT(5X17HSURFACE PARTICLES)
      M=0
      DO 14 N=1,NPART $ XX=SRP(N) $ YY=SLRP(N)
      K=XX/DR(1)+2 $ L=YY/DZ(1)+2 $ KM=K+(L-1)*KMAX
      IF(SUR(KM))15,14
15  M=M+1 $ GO TO (16,17,18),M
16  X1=XX $ Y1=YY $ GO TO 14
17  X2=XX $ Y2=YY $ GO TO 14
18  X3=XX $ Y3=YY $ M=0
19  FORMAT(9X,2E11.3,9X,2E11.3,9X,2E11.3)
      WOT 1B,19,X1,Y1,X2,Y2,X3,Y3
14  CONTINUE
      DO 459 L=LDMN,LDMX
      KTT=(L-1)*KMAX
      WOT 1B,200,(MESS(1),1=1,7)
200  FORMAT(2X,7A10)
      WOT 1B,470,Z(L),NCYCLE,TIME,MU,RHOA
470  FORMAT(3X,2HZ=,E14.6,3X,6HCYCLE=,16,3X,5HTIME=,E14.6,
12X,3H MU=,E14.6,2X,4HRHO=,E14.6)
      WOT 1B,460,L
      DO 466 K=KDMN,KDMX
      KP=K+KTT
      UPART=SUUA(KP)
      VPART=SLUA(KP)
      NP=NPRTA(KP)
      IF(NP)31,31,32
31  UPART=0.
      VPART=0.
      GO TO 33
32  UPART=UPART/NP
      VPART=VPART/NP
33  TEST=(U(KP),NE,0).UN.(V(KP),NE,0)
      IF(TEST)51,466
51  CONTINUE
      WOT 1B,461,K,ETA(KP),PSI(KP),U(KP),V(KP),P(KP),R(K)

```

CCC

```

369  NAPRT=0
      NBPRT=0
      FMASS=0.0
      ENKG=0.0
      ENGP=0.0
      UMOM=0.0
      VMOM=0.0
      PMIN=0.0
      PMA=0.0
      DCN=0.0
      KKK=.5*(2.+KBMN)+.5*ABSF(2.-KBMN)
      LLL=.5*(2.+LBMN)+.5*ABSF(2.-LBMN)
      DO 375 K=KKK,KDMX
      DO 375 L=LLL,LDMX
      KM=K+(L-1)*KMAX
      KB=KM-KMAX
      IF (FULL(KM).UN.SUR(KM))203,375
203  V1=.5*(V(KM)+V(KB))
      U1=.5*(U(KM)+U(KB-1))
      DM=RHOA*(ZB(L)-ZB(L-1))*(RB(K)**NEXPP-RB(K-1)**NEXPP)
      FMASS=FMASS+DM
      ENKG=ENKG+.5*DM*(U1**2+V1**2)
      ENGP=ENGP+Z(L)*DM*G
      UMOM=UMOM+DM*U1
      VMOM=VMOM+DM*V1
      NAPRT=NAPRT+NPRTA(KM)
      IF (ABSF(U1)-.001)41,41,40
41  IF (ABSF(V1)-.001)375,375,40
40  R1=R(K)
      Z1=Z(L)
      R2=R1+2.0*DT*U1*DR(1)/EPS1
      Z2=Z1+2.0*DT*V1*DZ(1)/EPS1
      CALL PLOTV(R1,Z1,R2,Z2)
375  CONTINUE

CCCC
      CHECK CONSERVATION OF MASS MOMENTUM AND ENERGY
      PI=3.14159265
      FMASS=FMASS*PI**NEXP
      ENKG=ENKG*PI**NEXP
      ENGP=ENGP*PI**NEXP
      VMOM=VMOM*PI**NEXP
      UMOM=UMOM*PI**NEXP
      CALL FRAME
      CALL SETCH(5,28,1,0,2,0,0)
      WRITE OUTPUT TAPE 100,27,PMIN,PMA,DCN
27  FORMAT(5HPMIN=,E11.3,5HPMA=,E11.3,3HDP=,E11.3)
      CALL SETCH(5,25,1,0,2,0,0)
      WRITE OUTPUT TAPE 100,28,FMASS,ENGP,ENKG
28  FORMAT(5HMASS=,E15.8,3HPE=,E15.8,3HKE=,E15.8)
      CALL SETCH(5,22,1,0,2,0,0)
      WRITE OUTPUT TAPE 100,29,DT,UMOM,VMOM
29  FORMAT(3HDT=,E11.3,6HUMOMT=,E15.8,6HVMOMT=,E15.8)
      CALL SETCH(5,19,1,0,2,0,0) $ WOT 100,201,(MESS(1),I=1,7)
201  FORMAT(7A10)
      RETURN
      END

```

DATA

*
1
1
1
1
1
QQ

SUBROUTINE
TTYMAIN

PROGRAMABC
(TAPE2, TAP

CALL
ASSIGN

CALL
MACMAIN

RETURN

C THIS ROUTINE CONTAINS COMMON AND PACKING FUNCTIONS.
 C NOTE: IN ORDER TO DO INTEGER ARITHMETIC WITH THE BYTES
 C MPRTB AND MPRTA (NUMBER OF TYPES OF PARTICLES ADDED TO A CELL)
 C IT IS NECESSARY TO SHIFT THEM LEFT 50 BITS THEN SHIFT THEM RIGHT 50 BITS. ON THE CDC6600 A LEFT SHIFT IS END AROUND WHILE A
 C RIGHT SHIFT REPLICATES THE LEFTMOST BIT. THE BYTES ARE AUTOMATICALLY RIGHT ADJUSTED IN A BLANK FIELD WHEN CALLED. THEREFORE FOR
 C A 10 BIT BYTE THE OPERATION (.SHL 50).SHR 50 FILLS THE 50 BITS
 C TO THE LEFT OF THE RIGHT ADJUSTED BYTE WITH THE VALUE OF THE
 C LEFT MOST BIT IN THE BYTE IE. THE SIGN OF THE BYTE.
 C THIS OPERATION IS NECESSARY BECAUSE PACKED INTEGERS MUST
 C SOMETIMES TAKE NEGATIVE VALUES
 C THE FUNCTIONS LURP THRU SLRP AND LUUA THRU SLUA MAKE IT POSSIBLE
 C TO PACK TWO FLOATING WORDS IN EACH WORD OF THE ARRAYS RP AND
 C UAVG. THE PACKED WORDS ARE GOOD TO 5 SIGNIFICANT FIGURES

CLICHE
CMAC

CALL
REGST

ENDCLICHE

CLICHE
PACK

MASKU=0000000007777
 777777B
 MASKL=77777777770000
 000000B
 LURP(N,B)=(RP(N).INT
 .MASKU).UN.(B.INT.MA
 SKL)
 LLRP(N,B)=(RP(N).INT
 .MASKL).UN.((B.INT.M
 ASKL).SHL 30)
 SURP(N)=RP(N).INT.MA
 SKL
 SLRP(N)=(RP(N).INT.M
 ASKU).SHL 30
 LUUA(J,B)=(UAVG(J).I
 NT.MASKU).UN.(B.INT.
 MASKL)
 LLUA(J,B)=(UAVG(J).I
 NT.MASKL).UN.((B.INT
 .MASKL).SHL 30)
 SUUA(J)=UAVG(J).INT.
 MASKL
 SLUA(J)=(UAVG(J).INT
 .MASKU).SHL 30

ENDCLICHE

CALL
UNLOAD

CALL
MACMAIN

JOBCHART

ROUTINE TTYMAI

CALLS : ASSIGN, MACMAIN

JOBCHART

ROUTINE MAIN.

CALLS : REGST , UNLOAD , MACMAIN

SUBROUTINE
MACMAIN

USE
CMAC

IA=3
IB=3
OPE=0.

C ARBITRARY BOUNDARY MODIFICATION OF
MARKER PARTICLE
C INCOMPRESSIBLE HYDRODYNAMIC CODE
C REF. LOS ALAMOS LABORATORY REPORT
LA-3425 (1966)
C LAWRENCE RADIATION LABORATORY REPO
RT UCRL 71595 (1969)
C JOURNAL OF COMPUTATIONAL PHYSICS,
VOL 4 NO. 4 (1969) P. 543
C FLOW IN, FLOW OUT, AND RIGID WALL
BOUNDARIES ALLOWED AT KMAX, LMAX
C FLOW OUT, AND RIGID WALL BOUNDARIE
S ALLOWED AT K=1, L=1
C PLANE FLOW OR CYLINDRICAL SYMMETRY
(ABOUT K LINE 1)

RETURN

IGOB(36)=ID1
IGOB(37)=ID2
IGOB(38)=ID3

CALL
CRTID

NTAL=0

362 2 NIXE

361 1

CALL
GEN

C NHC, NHD, NHE ARE TAPE LABELS

RETURN

367 = IF(NHC)

431

C COMPUTATION OF THE TRANSPORT TERMS
 IN THE MOMENTUM EQUATION
 C STORE IN ETA AND PSI

JOBCHART

ROUTINE MACMAI

CALLS : CRTID , GEN , RDX , EDIT , PRES , VEL57 , TRAJ , FLOWIN, SHIFT , REFLAG, 000TBF, SAM , EXIT

LABEL PAGE PAGES CONTAINING REFERENCES

LABEL	PAGE	PAGES CONTAINING REFERENCES
A10	14	13.
B10	3	3.
C10	13	
361	1	
431	1	
362	2	1.
367	2	1. 2. 2. 2. 2. 2. 2. 3.
364	2	
441	2	
442	2	2.
443	2	2.
444	2	2.
445	2	2.
446	3	2.
418	3	2.
22	3	
23	3	3.
24	3	3.
32	3	
33	3	3.
253	3	15. 15.
74	4	
11	4	
12	4	4.
42	4	
41	4	4.
43	4	4. 4.
13	4	
14	5	4.
44	5	
45	5	5.
46	5	5. 5.
47	5	
40	5	5.
49	5	
51	6	
52	6	6.
50	6	5.
25	6	
26	6	
27	6	
28	6	6.
76	6	6.
53	6	
55	6	
54	7	6. 6.
56	7	7.
82	7	
48	7	6. 6. 6. 6. 6. 7.

15	7							
16	7	7,						
58	7							
59	7	7,						
60	7	7,	7,					
17	8							
18	8	8,						
62	8							
63	8	8,						
64	8	8,	8,					
61	8							
57	8	8,						
66	9							
68	9							
69	9	9,						
67	9	9,						
29	9							
34	9							
35	9							
36	9	9,						
75	9	9,						
70	9							
72	10							
71	10	9,	10,					
73	10	10,						
79	10							
65	10	9,	9,	9,	9,	9,	10,	
81	10							
88	10							
89	11							
90	11	11,						
91	11	11,						
92	11							
93	11	11,						
94	11	11,						
87	11	10,						
96	11							
97	11							
98	11	11,						
99	12	11,						
101	12							
102	12	12,						
103	12	12,						
95	12	11,						
80	12	10,						
150	12							
351	13							
416	13							
415	14	13,	14,					
420	14							
419	14	14,						

203	14		
85	14	14,	
374	14		
372	15		
375	15		
373	15	15,	
371	15	15,	
254	15	14,	15,
433	15		
434	15	15,	

SUBROUTINE
GEN

C THIS ROUTINE IS THE GENERATOR FOR
THE THE ARBITRARY BOUNDARY
C VERSION THE THE MAC INCOMPRESSIBLE
HYDRODYNAMIC CODE.

USE
CMAC

USE
PACK

PI=3.14159265358979
IA=3
IB=3

RETURN

WOT3,202,(
MESS(1),1=

RETURN

WOT3,4,KMA
X,LMAX,NEX

RETURN

WOT3,86,SM
IN,EPS1,UB

RETURN

WOT3,86,GZ
,GR,MU,RHO

RETURN

WOT3,86,PA
MB,PGASZ,G

G=SQRTF(GZ**2+GR**2)
NEXPP=NEXP+1
KMAXZ=KMAX-1
KMAXZZ=KMAX-2
LMAXZ=LMAX-1
LMAXZZ=LMAX-2
NCYCLE=0
TIME=0.
NDIV=0
CUT=.0000001
NGRID=KMAX*LMAX
UM=MU

C GENERATE COORDINATES
C READ DR AND DZ

23

JOBCHART

ROUTINE GEN

CALLS : GENB , REFLAG

LABEL	PAGE	PAGES CONTAINING REFERENCES
-------	------	-----------------------------

A10	5	5.
23	1	2.
26	2	
21	2	
22	2	2. 2.
25	2	
27	2	
24	2	2.
28	2	
29	2	
30	2	
31	3	
70	3	
71	3	
39	3	
40	3	
32	3	5.
34	4	
604	4	4.
606	4	
605	4	4.
607	5	
609	5	
610	5	5.
608	5	5.
600	5	5. 5.
612	5	
36	5	5.
35	5	
91	5	4.
614	5	6.
94	6	
704	6	
705	6	6.
95	6	
92	6	6.
702	6	
706	6	
703	7	
701	7	6.
409	7	9.
410	7	
412	8	
411	8	7.
414	8	
420	8	7. 8. 8. 8.
421	8	
423	8	
422	8	8.

424	8				
440	8	8.	8.	8.	8.
442	9				
441	9				
408	9	7.	9.	9.	
431	9				
430	9	9.			
61	10				
60	10	10.			
62	10				
56	10	10.			
64	10				
63	10	10.			
65	10				
57	11	10.			
72	11

C COMPUTE BOUNDARY NORMALS FOR EACH
CELL CONTAINING A SEGMENT
C OF BOUNDARY AND SET CELL FLAGS. P
LACE NORMALS AT MID POINT OF
C SEGMENTS. CONVENTION IS THAT NORM
AL POINTS TO THE LEFT AS ONE
C MOVES FROM I TO I+1 ALONG SEGMENT.

22

I=0

501

I=I+1

C DETERMINE IF SEGMENT LIES ON TOP O
F A MESH LINE.

$$K = (.5 * (PTX(I+1) + PTX(I))) / DR(I) + 2$$

$$L = (1.0000001 * PTY(I)) / DZ(I) + 1$$

201 F IF ((PTY(I)
EQ.PTY(I+1)).INT.(P
TY(I).EQ.Z

T

200

202 ≤ IF (PTX(I+1)-PTX(I))

>

203

KM=L*KMAX+K
KB=KM-KMAX
DNORX(KM)=0.
DNORY(KM)=1.
PNORX(KM)=.5*(PTX(I)
+PTX(I+1))
PNORY(KM)=PTY(I)
OB(KM)=1
COR(KB)=1

617

202

KM=(L-1)*KMAX+K
KT=KM-KMAX
DNORX(KM)=0.
DNORY(KM)=-1.
PNORX(KM)=.5*(PTX(I)
+PTX(I+1))
PNORY(KM)=PTY(I)
COR(KT)=1
OB(KM)=1

617

201

L=(.5*(PTY(I+1)+PTY(I)))/DZ(I)+2
K=(1.0000001*PTX(I))/DR(I)+1

205 F IF ((PTX(I)
EQ.PTX(I+1)).INT.(P
TX(I).EQ.R

T

204

C GENERAL CASE, CELL SEGMENT NOT ON
C MESH LINE. COMPUTE AREA
C BOUNDED BY SEGMENT AND INTERIOR SI
C DES OF CELL. USE LINE INTEGRAL

205

C NOW DO THE PART OF THE LINE INTEGRAL
C THAT
C COMES FROM THE BOUNDARY SEGMENT.

C COMPUTE THE REST OF THE INTEGRAL

C NPRTB IS A BYTE WORD (PART OF FLAG
C) CONTAINING A POINTER FOR THE
C CELL WHICH CONTAINS THE MID POINT
C OF THE BOUNDARY SEGMENT
C INDICATING THE INDEX OF THE CELL T
C O BE USED IN

JOBCHART

ROUTINE GENB

CALLS : 000TBF, QUIT

LABEL	PAGE	PAGES CONTAINING REFERENCES
158	1	11.
120	1	1.
123	1	
122	1	1.
127	1	
129	2	
128	2	
130	2	2.
125	2	1.
133	2	
131	2	
132	2	2.
126	2	1.
134	2	2. 2.
137	2	
139	3	
138	3	
140	3	3.
135	3	2.
143	3	
141	3	
142	3	3.
136	3	2.
144	3	3. 3.
145	4	
146	4	
149	4	
148	4	4.
147	4	
150	4	
124	4	
153	4	5.
156	4	5.
160	4	
152	5	
154	5	
151	5	4. 5.
155	5	5.
23	5	
24	5	5.
21	5	
22	6	5.
501	6	5. 11.
200	6	
203	6	
202	6	6.
201	6	6.
204	6	
207	7	

206	7	7.			
617	7	6.	6.	7.	
205	7	6.			
208	7	7.			
210	7				
211	7	7.			
209	8	7.			
213	8				
212	0	0.			
215	8				
216	8				
217	8	8.			
214	8	8.			
300	8	7.	7.	8.	8.
301	9	8.			
303	9				
304	9				
305	9	9.			
302	9	9.			
306	9				
307	9	9.			
399	9	9.	9.	9.	
308	9				
310	10				
309	10	9.			
311	10				
314	10				
398	10	10.			
400	10	8.	10.		
402	10				
401	10	10.			
500	11	7.			
502	11				
49	11				
50	11				
48	11	11.	11.		
157	11	1.			

SUBROUTINE
PRES

USE
CMAC

IB=3

C THIS ROUTINE SOLVES POISSON EQUATION FOR THE PRESSURE
C BY SUCCESSIVE OVER RELAXATION NOT E.
C $U = \eta - DP/DR$, $V = \psi - DP/DZ$
C WHERE η , ψ ARE CONSTANTS DURING ITERATION
C THEREFORE $P(N+1) = P(N) - RELAX * DIVERGENCE(VELOCITY)$
C GENERATES SOR PROVIDED U AND V ARE RECOMPUTED EACH TIME THE
C CELL PRESSURE IS ADJUSTED
C RELAX PARAMETER HAS MAXIMUM LIMIT FOR STABILITY AND OPTIMUM
C VALUE FOR MOST RAPID CONVERGENCE
C $RELAX MAX = .5 * \rho * \Delta X ** 2 / \Delta T$
C OPTIMUM VALUE IS A FUNCTION OF GEOMETRY ETC. USUALLY VERY CLOSE TO
C TO MAX VALUE. SEE ANY SOR TEXT.

NEX=0

CALL
VEL

33

M=0

424

N=0

188

N=N+1
ENORM=0.
PNORM=0.

DO 423
L=2,LMA

KTT=L*KMAX

DO 423
K=2,KMA

KP=K
LP=L
NT=KTT+KP
NM=NT-KMAX
NB=NM-KMAX

IF (FULL(NM
) .INT. (.NOT. OB(NM)))

F 22

21

DIV=(U(NM)*RB(K)**NEXP-U(NM-1)*RB(K-1)**NEXP)/ (R(K)**NEXP*DR
(1))
DIV=DIV*(V(NM)-V(NB))/DZ(1)

24

C IN BOUNDARY CELLS REPLACE DIV(V) BY
Y (BOUNDARY NORMAL) DOT

```

C      IF P NEGATIVE IN BOUNDARY CELL SET
P=0 TO ALLOW FLOW SEPARATION.

```

JOBCHART

ROUTINE PRES

CALLS : VEL , VELD

LABEL	PAGE	PAGES CONTAINING REFERENCES
33	1	
424	1	
188	1	6.
21	1	
22	2	1.
23	2	
52	2	
53	2	2.
55	2	2.
54	2	2.
56	2	2.
58	2	
57	2	2.
59	2	2.
61	3	
60	3	3.
62	3	3.
64	3	
63	3	3.
65	3	3.
66	3	
67	3	3.
68	3	3.
71	3	
70	4	3.
72	4	3.
74	4	
73	4	4.
75	4	4.
76	4	
77	4	4.
78	4	4.
80	4	
79	4	4.
81	4	4.
24	4	1.
93	4	
91	5	
92	5	5.
422	5	2.
423	5	
183	5	
20	5	
184	5	
425	5	
406	6	
405	6	6.
426	6	

427
189

6
6

6.

SUBROUTINE
TRAJ

USE
CMAC

USE
PACK

C THIS ROUTINE MOVES THE MARKER PART
ICLES BY USUAL MAC
C WEIGHTING FORMULA. R AND Z COORDI
NATES OF PARTICLE PACKED IN RP.

IB=3
IA=3

DO 317
N=1,NGR

UAVG(N)=0.0
MPRTA(N)=0

317

N=0

436

N=N+1

449

437 > IF (N-NPART)

450

C UNPACK RP

XR=SRP(N)
XZ=SLRP(N)
KP=(XP+DR(1))/DR(1)+
LP=(XZ+DZ(1))/DZ(1)+
NT=LP*KMAX+KP
NM=NT-KMAX
NB=NM-KMAX
DLLR=RB(KP)-RB(KP-1)
DLLZ=ZB(LP)-ZB(LP-1)

53 > IF (X7-7(LP))

52

LB=LP-1

69

53

LB=LP

69

KT=LB*KMAX+KP
KM=(LB-1)*KMAX+KP
SX=(R(KP)-XR)/DLLR
SY=(ZB(LB)-XZ)/DLLZ

JOBCHART

ROUTINE TRAJ

LABEL	PAGE	PAGES CONTAINING REFERENCES
A10	4	4.
A20	4	4.
A30	5	4. 4.
A40	5	4. 4. 4. 4.
A50	5	5.
317	1	
436	1	6.
449	1	5.
450	1	
52	1	
53	1	1.
69	1	1.
55	2	
54	2	2.
56	2	2.
58	2	
57	2	2.
59	2	2.
61	2	
60	2	2.
62	2	2.
64	2	
63	2	2.
65	2	2.
66	3	
67	3	3.
68	3	3.
71	3	
70	3	3.
72	3	3.
74	3	
73	3	3.
75	3	3.
76	3	
77	3	3.
78	3	3.
80	4	
79	4	4.
81	4	4.
45	5	
346	5	5. 5.
89	5	
98	5	
46	6	5.
437	6	1.

```
NPART=NPART+1  
RP(NPART)=RP(N)  
NOIV=NPART  
RP(N)=LURP(N,X)  
RP(N)=LLRP(N,Y)  
NPRTA(L)=NPRTA(L)+1
```

JOBCHART

ROUTINE FLOWIN

LABEL	PAGE	PAGES CONTAINING REFERENCES
2	1	
5	1	
4	1	1.
3	1	1.
7	1	
8	1	1.
6	1	1. 1. 1.
1	2	1. 1. 1.

JOBCHART

ROUTINE SHIFT

LABEL	PAGE	PAGES CONTAINING REFERENCES
220	1	
44	1	
50	1	
48	1	1. 1.

```

C   THE FIRST TIME CELLS PRESSURE IS T
O BE COMPUTED LOAD CELL
C   PRESSURE WITH AVERAGE OF ITS NEIGH
BORS.

```

JOBCHART

ROUTINE REFLAG
CALLS : VEL55

LABEL PAGE PAGES CONTAINING REFERENCES

LABEL	PAGE	PAGES CONTAINING REFERENCES
64	1	
21	1	
24	1	
25	1	
26	1	1.
27	1	
28	2	1.
29	2	
30	2	2.
31	2	
22	2	1.
32	2	
33	2	
34	2	
35	2	2.
36	2	
37	2	2.
38	3	
39	3	3.
40	3	
20	3	1. 2. 2. 2. 2. 3.
42	3	
43	3	
46	3	
47	3	
48	3	
45	4	3. 3. 3.
44	4	3.
87	4	
50	4	
51	4	
52	4	
53	4	4.
54	4	
55	4	
56	4	4.
57	5	
58	5	
59	5	5.
60	5	
61	5	
62	5	5.
66	5	
67	5	5.
41	5	3. 4. 4. 4. 4. 4. 4. 5. 5. 5.
126	6	
120	6	
122	6	6.
121	6	6. 6.

123	6				
125	6	6.			
124	6	6.	6.		
104	6				
103	7	6.			
92	7				
93	7				
94	7	7.			
95	7				
96	7	7.			
97	7				
98	7	7.			
99	7				
101	7	7.			
102	7				
91	7	6.	6.	7.	7.
90	7				
76	8				
88	8				
73	8	8.	8.		

SUBROUTINE
VEL

USE
CMAC

C THIS ROUTINE COMPUTES ALL OF THE CELL VELOCITIES. THE
C PRESSURE GRADIENTS ARE CALCULATED FOR BOUNDARY PRESSURES
C LOCATED AT THE MID POINTS OF THE LAGRANGE BOUNDARY SEGMENTS.
C APPROXIMATE LINEAR INTERPOLATION USED. THIS ROUTINE AND VELP ARE
C THE SAME EXCEPT VELP COMPUTES ONLY A SINGLE CELL.

DO 192
L=2,LMA

KTT=L*KMAX

DO 192
K=2,KMA

KT=K+KTT
KM=KT-KMAX
KB=KM-KMAX
KTZ=KT+KMAX
TEST=FULL(KM).UN.(SUR(KM).INT.(.NOT.COR(KM)))

C BOUNDARY CONDITIONS ON VELOCITIES
AT EDGE OF MESH

JOBCHART

ROUTINE VEL

CALLS : VEL55

LABEL	PAGE	PAGES CONTAINING REFERENCES
20	1	1.
36	1	
37	1	
42	1	
21	2	1.
70	2	
71	2	2.
72	2	
73	2	2.
88	2	
89	2	2.
91	2	
90	2	2.
92	2	2.
93	2	
94	3	2.
96	3	
95	3	3.
97	3	3.
25	3	1. 1. 1.
24	3	
22	3	
45	3	
23	3	3.
74	3	
75	3	3.
76	4	
77	4	4.
78	4	
79	4	4.
81	4	
80	4	4.
82	4	4.
83	4	
84	4	4.
86	4	
85	5	4.
87	5	4.
192	5	1. 3. 3. 3.
27	5	
28	5	
29	5	5.
32	5	
31	5	5. 5.
33	5	
34	6	5.
35	6	
26	6	5. 5. 6.
50	6	

51	6	6.		
53	6			
54	6	6.		
700	6			
52	6	6.	6.	6.
55	7			
56	7	7.		
58	7			
59	7	7.		
710	7			
193	7	7.	7.	7.
60	7			
61	7	7.		
63	7			
64	8	7.		
720	8			
62	8	7.	8.	8.
65	8			
66	8	8.		
67	8			
68	8	8.		
730	8			
194	8	8.	8.	8.

JOBCHART

ROUTINE VELP
CALLS : VEL55P

LABEL	PAGE	PAGES CONTAINING REFERENCES
20	1	1.
36	1	
37	1	
42	1	
21	1	1.
70	1	
71	2	1.
72	2	
73	2	2.
88	2	
89	2	2.
91	2	
90	2	2.
92	2	2.
93	2	
94	2	2.
96	3	
95	3	2.
97	3	3.
25	3	1. 1. 1.
24	3	
22	3	
45	3	
23	3	3.
74	3	
75	3	3.
76	4	
77	4	4.
78	4	
79	4	4.
81	4	
80	4	4.
82	4	4.
83	4	
84	4	4.
86	4	
85	5	4.
87	5	4.
192	5	1. 3. 3. 3.
27	5	
28	5	
29	5	5.
32	5	
31	5	5. 5.
33	5	
34	5	5.
35	5	
26	6	5. 5. 5.

JOBCHART

ROUTINE VEL55

LABEL	PAGE	PAGES CONTAINING REFERENCES
2	1	
3	1	
4	1	1.
5	1	
6	2	1.
7	2	
8	2	2.
9	2	
10	2	2.
1	2	1. 1. 2. 2. 2.

JOBCHART

ROUTINE VEL55P

LABEL	PAGE	PAGES CONTAINING REFERENCES
-------	------	-----------------------------

2	1	
3	1	
4	1	1.
5	1	
6	2	1.
7	2	
8	2	2.
9	2	
10	2	2.
1	2	1. 1. 1. 2. 2.

LABEL	PAGE	PAGES CONTAINING REFERENCES
52	1	
53	1	1.
69	1	1.
55	1	
54	1	1.
56	1	1.
58	1	
57	1	1.
59	2	1.
61	2	
60	2	2.
62	2	2.
64	2	
63	2	2.
65	2	2.
66	2	
67	2	2.
68	2	2.
71	2	
70	3	2.
72	3	3.
74	3	
73	3	3.
75	3	3.
76	3	
77	3	3.
78	3	3.
80	3	
79	3	3.
81	3	3.

SUBROUTINE
VEL57

USE
CMAC

USE
PACK

C THIS ROUTINE TAKES CARE OF CELLS I
N WHICH A FREE SURFACE AND
C AN ARBITRARY BOUNDARY INTERSECT.
PROVISION ADDED FOR LATER
C ADDITION OF MOVING BOUNDARY, THERE
FORE MAKE CERTAIN UBND
C AND VBND ARE ZERO

DO 20
J=1,NGR

SNC(J)=SUR(J),INT,(,
NOT,COR(J))

20

DO 92
L=2,LMA

KT=L*KMAX

DO 92
K=2,KMA

KT=KT+K
KM=KT-KMAX
KB=KM-KMAX

C DETECT INTERSECTION.

TEST=OB(KM),INT,ARB(
KM),INT,(,NOT,OK(KM)
,INT,(,NOT,EMP(KM))

C CELL CONTAINS INTERSECTION. DETEC
T CASE WHERE FLUID IS
C SEPARATING FROM THE BOUNDARY.

93

PNX=PNORX(KM)
PNY=PNORY(KM)
DNX=DNORX(KM)
DNY=DNORY(KM)

CALL
VEL56

UP=UP-UBND
VP=VP-VBND

JOBCHART

ROUTINE VEL57

CALLS : VEL56

LABEL	PAGE	PAGES CONTAINING REFERENCES										
20	1											
93	1	1,										
94	2	2,										
103	2											
116	2											
117	2	2,										
118	2	2,										
105	2											
107	3											
108	3	3,										
104	3	2,										
109	3											
126	3											
127	3	3,										
128	3	3,										
110	4											
111	4											
112	4	4,										
106	4	2,	4,	5,	5,							
100	4	3,										
129	4											
131	5											
130	5	4,										
132	5											
133	5											
134	5	5,										
115	5											
92	6	1,	2,	3,	3,	4,	4,	4,	5,	5,	5,

JOBCHART

ROUTINE SAM

CALLS : UNLOAD, ASSIGN, EXIT

LABEL	PAGE	PAGES CONTAINING REFERENCES		
37	1.			
28	1	2.	2.	
20	1	1.		
21	1	1.		
36	1	1.		
22	1	1.		
12	1	1.	1.	
11	2			
2	2	2.		
7	2	2.	3.	3.
23	2	2.	2.	
24	2	2.		
26	2			
25	2	2.		
27	2			
3	2	2.		
40	2	3.		
41	3			
33	3			
34	3	3.		
35	3	3.		
29	3	3.	3.

JOBCHART

ROUTINE RDX

CALLS : ASSIGN, UNLOAD, EXIT

LABEL	PAGE	PAGES CONTAINING REFERENCES
32	1	1.
34	1	1.
37	1	2.
43	1	
42	1	1.
35	2	1., 1.
50	2	2.
51	2	2.

JOBCHART

ROUTINE EDIT

CALLS : 000TBF , FRAME , SETCH , MAPX , LINE , POINT , PLOTV

LABEL	PAGE	PAGES CONTAINING REFERENCES
-------	------	-----------------------------

410	1	
15	1	
16	1	
17	1	
18	2	
14	2	1, 1, 1.
31	2	
32	2	2.
33	2	2.
51	2	
466	2	2.
38	3	
50	3	
464	3	3.
459	3	3.
374	3	4.
62	4	
61	4	4.
368	4	
370	4	
369	5	4.
203	5	
41	5	
40	5	5.
375	5	5., 5.

Distribution: Internal

J. Viecelli (15)

TID (15)

T Division Files (3)

External

TIC (2)
Oak Ridge
TN

Argonne Code Center
Att'n. M. Butler
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

University of South Wales
Attn: Mr. L.G. Nhan
Mechanical and Industrial Engineering
P.O. Box 1
Kensington, South Wales Australia 2033

NOTICE

"This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research & Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights."

Technical Information Department

LAWRENCE LIVERMORE LABORATORY

University of California | Livermore, California | 94550