58 cm

MODCONV : SIMULATION OF MODERATOR

MODCONV : A COMPUTER CODE TO PREDICT THE LOCAL VELOCITIES AND TEMPERATURES INSIDE A MODERATOR

By

YUK-TONG CHAN B.Sc.(Hons)

A Project Report (Off-Campus) Part B

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Master of Engineering

McMaster University

April 1980

ABSTRACT

MODCONV is a FORTRAN Computer code using the primitive form of the conservation equations for natural convection in porous medium to calculate the velocity and temperature distribution in the moderator of a typical CANDU reactor under normal and shut down operations, as well as postulated LOCA for which the rupture in the system is external to the core. Appropriate velocities are imposed at the boundary walls so that the free convection equations set can be used to model a mixed convection problem. Purely explicit, donor cell finite differencing scheme is used. A heat exchanger model using NTU method is also included to give a realistic inflow temperature for the moderator. The heat load distribution is calculated according to a channel power map, but can be redefined through input.

ACKNOWLEDGEMENTS

The author wishes to express his gratitude to A. Chan, C. Choo, W.I. Midvidy and R.E. Pauls for their suggestions and encouragement. Appreciation is also expressed to Ontario Hydro for the computer facilities provided throughout the course of this research.

TABLE OF CONTENTS

LIST OF FIGURES

Nomenclatures:

A_v - specific surface area (m^2)

C - specific heat capacity (kJ/kg^oC)

D_p - characteristic length of the particles (m)

g - gravitational acceleration (m/sec^2)

K - permeability of the porous medium (m^2)

P - pressure (N/m^2)

P_l - pressure fluctuation (N/m^2)

r - radius of the cylinders (m)

T - temperature (^oC)

T_e - reference temperature (^oC)

v - filtration velocity (m/sec)

β - volumetric expansion coefficient ($/^oC$)

λ - thermal conductivity ($kJ/m\ sec^oC$)

ν - kinemetic viscosity (m^2/sec)

ρ - density (kg/m^3)

ϕ - porosity

∇ - $i\ \partial/\partial x + j\ \partial/\partial y$ (m^{-1})

I INTRODUCTION

The goal of developing MODCONV is to investigate the flow patterns and temperature distribution inside the moderator. The code can handle the case of normal and shut down operations, as well as those postulated LOCA for which the system rupture is external to the core.

Numerical scheme used is similar to the one described by A. Chan (1) for a pure fluid in a rectangular enclosure without flow boundaries. A porous region surrounded by a pure fluid with flow boundaries is used to simulate the moderator. The basic equations set used is the primitive form of the conservation equations for natural convection in porous medium. A heat exchanger model using NTU method is included to calculate transient effects. Local heat generation is distributed according to a channel power distribution map, but can be rewritten either through tape or input cards. The heat generation and temperature in different locations can also be monitored during power transient calculations.

II MATHEMATICAL FORMULATIONS

For natural convection flows with small density changes the conservation equations may be simplified by the Boussinesq approximation. The fluid density is considered constant except in the buoyancy force term which drives the natural convection. For a fluid with a linear density variation caused by temperature changes alone, and for a homogeneous and isotropic porous medium, the conservation equation of the mass, momentum and energy may then be written as in Combarnous (2):

$$\nabla \left(\frac{V}{\phi}\right) = 0 \qquad 2\text{-}1$$

$$\frac{\partial V}{\partial t} = -(V\cdot\nabla)\frac{V}{\phi} - \frac{\phi}{\rho}\nabla P' - \beta \phi g (T - T_e) - \left(\phi \frac{\nu}{K} V + B V|V|\right) + \nu \phi \nabla^2\left(\frac{V}{\phi}\right) \qquad 2\text{-}2$$

$$\phi \rho C \frac{\partial T}{\partial t} = \nabla\cdot(\lambda \nabla T) - \nabla(\rho C V T) + Q''' \qquad 2\text{-}3$$

For the sake of completeness, both the Darcy and viscous frictional drag terms are included in the momentum equations. In the pure fluid region, the Darcy drag term is zero and only the viscous drag term is effective. Similarly, the viscous drag term is relatively small as compare to the Darcy drag term in the porous region.

The initial conditions for temperature and velocity must be specified. For the finite difference formulation described later this amounts to specifying the initial velocity and temperature at each mesh point or cell.

In our particular problem, we have considered rigid, no slip walls, except at the inlet and outlet locations which we imposed the desired velocities and temperatures.

Concerning the permeability which is also required, it is calculated by the Blake - kozency relationship from Bird (3):

$$K = \frac{D_p^2 \phi^3}{150 (1-\phi)^2} \qquad 2-4$$

where D_p is the Characteristic length of the constituting particles.

In general

$$D_p = \frac{6}{a_v} \qquad 2-5$$

and the specific surface, a_v, for the cylindrical particles is ^{2}/radius. Then the permeability for a porous bed which consists of cylindrical particles is

$$0.06 r^2 \frac{\phi^3}{(1-\phi)^2} \qquad 2\text{-}6$$

The second term of the Darcy's frictional drag is obtained from Burke-Plummer equation for turbulent flow, where B is

$$\frac{1.75 \cdot (1-\phi)}{D_p \quad \phi^3} \qquad 2\text{-}7$$

Note that for high rate of flow the Burke-Plummer equation becomes important while for low rate of flow the Blake-Kozeny equation becomes important.

III MODEL OF THE MODERATOR SYSTEM

A typical CANDU moderator system is shown in Figure 1. Our model involves the calandria and the heat exchangers system. The cross section of the calandria is idealized as in Figure 2. The heat exchangers are treated as a separate system. Detailed formulation and computation procedures for the heat exchangers model can be found in Appendix B.

To simplify the problem we neglect axial effects in the calandria. For this reason, we can model the velocity and temperature field with two dimensional transient convective flow for porous medium in an enclosure. The center part of the calandria, where the fuel channels are located, is simulated by a cylindrical - shaped porous bed, while the rest of the calandria is treated as pure fluid. Conductivity, viscosity and heat capacity of the heavy water for the conservation equations are taken at each cell temperature, while the density in the momentum equation is taken at a fixed reference temperature. Inlet and outlet flows are imposed through he boundary conditions at each time step.

The total moderator heat load under normal operation includes neutronic heating of the moderator as well as "conventional" heat transfer from fuel channel. Heat load is distributed in a similar fashion as the channel power. For a reactor shut down or accident case, the total and local heat load can be changed with respect to time.

IV NUMERICAL TECHNIQUE

The conservation equations 2-1 to 2-3 are written in finite difference form for cells of the type shown in Fig 3. Subscripts i, j denote quantities at the cell center. Subscripts i+1/2, j+1/2, denote quantities on the right hand, and upper cell faces respectively. The finite difference form of the equations used for our calculations use donor cell or fully upsteam formulation for the advective terms. The essential feature of the donor-cell differencing scheme is that the transport of the momentum and energy components is advected only in the direction of the velocity.

Mass conservation equation:

$$D = \left[U^{n+1}_{i+\frac{1}{2},j} \left/ \left(\phi_{i+1,j} + \phi_{i,j} \right) - U^{n+1}_{i-\frac{1}{2},j} \right/ \left(\phi_{i,j} + \phi_{i-1,j} \right) \right] / \Delta X_{ij}$$

$$+ \left[V^{n+1}_{i,j+\frac{1}{2}} \left/ \left(\phi_{i,j+1} + \phi_{i,j} \right) - V^{n+1}_{i,j-\frac{1}{2}} \right/ \left(\phi_{i,j} + \phi_{i,j-1} \right) \right] / \Delta Y_{ij} = 0 \qquad 4\text{-}1$$

Momentum equations:

$$U^{n+1}_{i+\frac{1}{2},j} = U^{n}_{i+\frac{1}{2},j} + \Delta t \cdot \left[\left(P''^{n}_{i,j} - P''^{n}_{i+1,j} \right) / \Delta X_{ij} + PXI \cdot (VISX -$$

$$\text{COEX}\cdot U^{n}_{i+\frac{1}{2}j} - \text{COX}\cdot|U^{n}_{i+\frac{1}{2}j}|U^{n}_{i+\frac{1}{2}j}) - \text{FUX} - \text{FUY} \Big] \qquad 4\text{-}2$$

$$V^{n+1}_{i\,j+\frac{1}{2}} = V^{n}_{i\,j+\frac{1}{2}} + \Delta t \Big[(P^{n}_{i\,j} - P^{n}_{i\,j+1}) / \Delta Y_{ij} + \text{PY1}\cdot(1/2\,\beta g\,($$

$$T_{i\,j+1} + T_{i\,j} - 2\,T_{o}) + \text{VISY} - \text{COEY}\cdot V_{ij+\frac{1}{2}} - \text{COY}\cdot|V_{i\,j+\frac{1}{2}}|V_{i\,j+\frac{1}{2}}) - \text{FVX} - \text{FVY} \Big]$$

$$4\text{-}3$$

where,

$$\text{FUX} = ((U_{i+1\frac{1}{2}j}/\text{PX2} - U_{i+\frac{1}{2}\,j}/\text{PX1})\cdot(U_{i+\frac{1}{2}\,j} - |U_{i+1_2\,j}|) +$$

$$(U_{i+\frac{1}{2}j}/\text{PX1} - U_{i-\frac{1}{2}j}/\text{PX0})\cdot(U_{i+\frac{1}{2}j} + |U_{i+\frac{1}{2}j}|)/2./\Delta X$$

$$\text{FUY} = ((2.U_{i+\frac{1}{2}j+1}/(\phi_{i+1\,j-1} + \phi_{i\,j-1}) - U_{i+\frac{1}{2}\,j}/\text{PX1})\cdot(\text{VX} - |\text{VX}|) +$$

$$(U_{i+\frac{1}{2}\,j}/\text{PX1} - 2.U_{i+\frac{1}{2}\,j-1}/(\phi_{i+1\,j-1} + \phi_{i\,j-1}))(\text{VX} + |\text{VX1}|))/2./\Delta Y$$

$$\text{FVX} = ((2.V_{i+1\;\,j+\frac{1}{2}}/(\phi_{i+1\,j+1} + \phi_{i+1\,j}) - V_{i\;\,j+\frac{1}{2}}/\text{PY1})(\text{UY} - |\text{UY1}|) +$$

$$(V_{i\,j+\frac{1}{2}}/\text{PY1} - 2.V_{i-1\;\,j+\frac{1}{2}}/(\phi_{i-1\;\,j} + \phi_{i-1\;\,j+1}))\cdot(\text{UY} + |\text{UY1}|))/2./\Delta X$$

$$VX = (V_{i\ j+1/2} + V_{i\ j-1/2} + V_{i+1\ j+1/2} + V_{i+1\ j-1/2})/4.$$

$$UY = (U_{i-1/2\ j} + U_{i-1/2\ j+1} + U_{i+1/2\ j} + U_{i+1/2\ j+1})/4.$$

$$VISX = CX\cdot((U_{i+1/2\ j}/PX2 - 2.U_{i+1/2\ j}/PX1 + U_{i-1/2\ j}/PX0)/\Delta X^2 +$$

$$2.(U_{i+1/2\ j+1}/(\phi_{i+1\ j+1} + \phi_{i\ j+1}) - U_{i+1/2\ j}/PX1 + U_{i+1/2\ j-1}/(\phi_{i+1\ j-1}$$

$$+ \phi_{i\ j-1}))/\Delta Y^2)$$

$$VISY = CY\cdot(2.(V_{i+1\ j+1/2}/(\phi_{i+1\ j+1} + \phi_{i+1\ j}) - V_{i\ j+1/2}/PY1 + V_{i-1\ j+1/2}/$$

$$(\phi_{i-1\ j+1} + \phi_{i-1\ j}))/\Delta X^2 + (V_{i\ j+1\frac{1}{2}}/PY2 - 2.V_{i\ j+1/2}/PY1 +$$

$$V_{i\ j-1/2}/PY0)/\Delta Y^2)$$

$$PX0 = (\phi_{i-1\ j} + \phi_{i\ j})/2.$$

$$PX1 = (\phi_{i\ j} + \phi_{i+1\ j})/2.$$

$$PX2 = (\phi_{i+1\,j} + \phi_{i+2\,j})/2.$$

$$CX = (\mu_{i+1\,j} + \mu_{i\,j})/2./\rho$$

$$COEX = 16.67 \cdot CX \cdot (1. - PX1)^2 / PX1^3 / RAD^2$$

$$COX = 0.583 \cdot (1. - PX1)/PX1^3/RAD$$

Energy equation:

$$T_{ij}^{n+1} = T_{ij}^{n} + \Delta t / \phi_{i\,j} /(\rho C)_{ij}^{n} \cdot (TSX + TSY - TUX - TVY + Q''') \qquad 4.4$$

where

$$TUX = ((U_{i+\frac{1}{2}j} + U_{i-\frac{1}{2}\,j} - |U_{i+\frac{1}{2}j} + U_{i-\frac{1}{2}j}|) \cdot (T_{i+1\,j} \cdot (\rho C)_{i+1\,j} -$$

$$T_{i\,j} \cdot (\rho C)_{i\,j}) + (U_{i+\frac{1}{2}\,j} + U_{i-\frac{1}{2}\,j} + |U_{i+\frac{1}{2}\,j} + U_{i-\frac{1}{2}\,j}|) \cdot (T_{i\,j} \cdot (\rho C)_{i\,j} -$$

$$T_{i-1\,j} (\rho C)_{i-1\,j}))/4./\Delta X$$

$$TVY = ((V_{ij+1/2} + V_{ij-1/2} - |V_{ij+1/2} + V_{ij-1/2}|)\cdot(T_{ij+1}\cdot(\rho C)_{ij+1} -$$

$$T_{ij}\cdot(\rho C)_{ij}) + (V_{ij+1/2} + V_{ij-1/2} + |V_{ij+1/2} + V_{ij-1/2}|)\cdot(T_{ij}\cdot(\rho C)_{ij} -$$

$$T_{ij-1}(\rho C)_{ij-1}))/4./\Delta Y .$$

$$TSX = ((\lambda_{i+1\,j} + \lambda_{i\,j})\cdot(T_{i+1\,j} - T_{i\,j}) - (\lambda_{i\,j} + \lambda_{i-1\,j})\cdot$$

$$(T_{i\,j} - T_{i-1\,j}))/2./\Delta X^2 .$$

$$TSY = ((\lambda_{ij+1} + \lambda_{i\,j})\cdot(T_{ij+1} - T_{i\,j}) - (\lambda_{i\,j} + \lambda_{i\,j-1})\cdot$$

$$(T_{i\,j} - T_{i\,j-1}))/2./\Delta Y^2 .$$

To impose the necessary boundary conditions, the fluid is considered to be surrounded by a single layer of fictitious cells in which the variables are specified.

i) typical rigid, no slip wall cell

$$U_{1\frac{1}{2},j} = 0.$$

$$V_{1\,j+\frac{1}{2}} = -V_{2\,j+\frac{1}{2}}$$

ii) At the inlet flow cells

$$U_{1\frac{1}{2}j} = U_I$$

$$V_{1\,j+\frac{1}{2}} = V_{2\,j+\frac{1}{2}} = V_I$$

$$V_{1\,j-\frac{1}{2}} = V_{2\,j-\frac{1}{2}} = V_I$$

where U_I and V_I are the X^{th} and y^{th} component of the inflow velocity.

iii) At the outlet flow cell

$$V_{i\,1\frac{1}{2}} = -V_o$$

where V_o is the outflow velocity.

iv) All wall cells are adiabatic except at the two inlets

$$T_{1\,j} = T_{2\,j}$$

The temperatures of the corner boundary cells are set to the average of the temperatures of the three adjacent cells.

v) Isothermal, constant temperature wall cells at the two inlets

$$T_{1\,j} = 2.\,T_{boundary} - T_{2\,j}$$

V NUMERICAL STABILITY AND COMPUTATION TIME

The finite difference equations are in the donor cell form. Numerical stability is obtained provided the fluid is not permitted to cross more than one cell in one time step. Thus, the time increment must satisfy the usual courant number criterion for purely explicit schemes:

$$C_x = \frac{U \Delta t}{\phi \Delta X} < 1.0$$

$$C_y = \frac{V \Delta t}{\phi \Delta X} < 1.0$$

In our problem, the maximum velocity is found to be the inlet flow velocity. Therefore, the optimized time increment is calculated with the above equations at the inlet cell. For a flow of 0.89 m^3/sec at 76^o, the maximum time increment is about 0.12 seconds.

The problem computation time required per time increment (cycle) depends, in general, on i) the number of cells, ii) the cell size, iii) the time step size, and iv) the convergence criterion set for the velocity field iteration to satisfy the continuity equation. The cell size

and time step size are interrelated by the stability conditions.

In our problem, we found that it took about 0.055 seconds for each iteration in a 33 x 32 meshes (0.27384 x 0.28575 m^2 each) calculation, thus, it took 0.572 + 0.055 x no. of iterations seconds for each time step calculations, in which 0.002 seconds was consumed in the heat exchangers model. We have chosen the convergence criterion to be 4 x 10^{-4}.

VI CALCULATION PROCEDURES

i) Compute estimates for the new velocity field from the momentum equations using previous time step values for all quantities on the right hand side of the equation 4-2, 4-3, this is done for all cells.

ii) Adjust the new velocity field iteratively to satisfy the mass conservation equation by changing the cell pressures. If the divergence of a cell is negative, this corresponds to a net flow of mass into the cell, the cell pressure is then increased to eliminate the net inflow. Likewise, the cell pressure can be decreased to eliminate net outflow if D is positive. If the pressure of one cell is adjusted, then the adjacent cells are also affected. Therefore, the pressure adjustment must be done interatively. The pressure change required to make D equal to zero is:

$$\Delta P'' = - \frac{D}{2.\Delta t\,(A + B)}$$

where

$$A = \frac{1}{\Delta X^2}\left(\frac{1}{\phi_{i+1\,j} + \phi_{i\,j}} + \frac{1}{\phi_{i\,j} + \phi_{i-1\,j}}\right)$$

$$B = \frac{1}{\Delta Y^2}\left(\frac{1}{\phi_{i\,j+1} + \phi_{i\,j}} + \frac{1}{\phi_{i\,j} + \phi_{i\,j-1}}\right)$$

The new pressure and velocity field components after each iteration are given by

$$P''_{i\,j} = P''_{i\,j} + \Delta P''$$

$$U_{i\pm\frac{1}{2}j} = U_{i\pm\frac{1}{2}j} \pm \Delta t \cdot \Delta P''/\Delta X$$

$$V_{ij\pm 1/2} = V_{ij\pm 1/2} \pm \Delta t \, \Delta \tilde{P} / \Delta Y$$

$\Delta \tilde{p}$ is obtained when the above u and v are substituted into the continuity equation.

iii) When the new velocity field has been calculated, the necessary velocity boundary conditions are imposed at the appropriate wall cells.

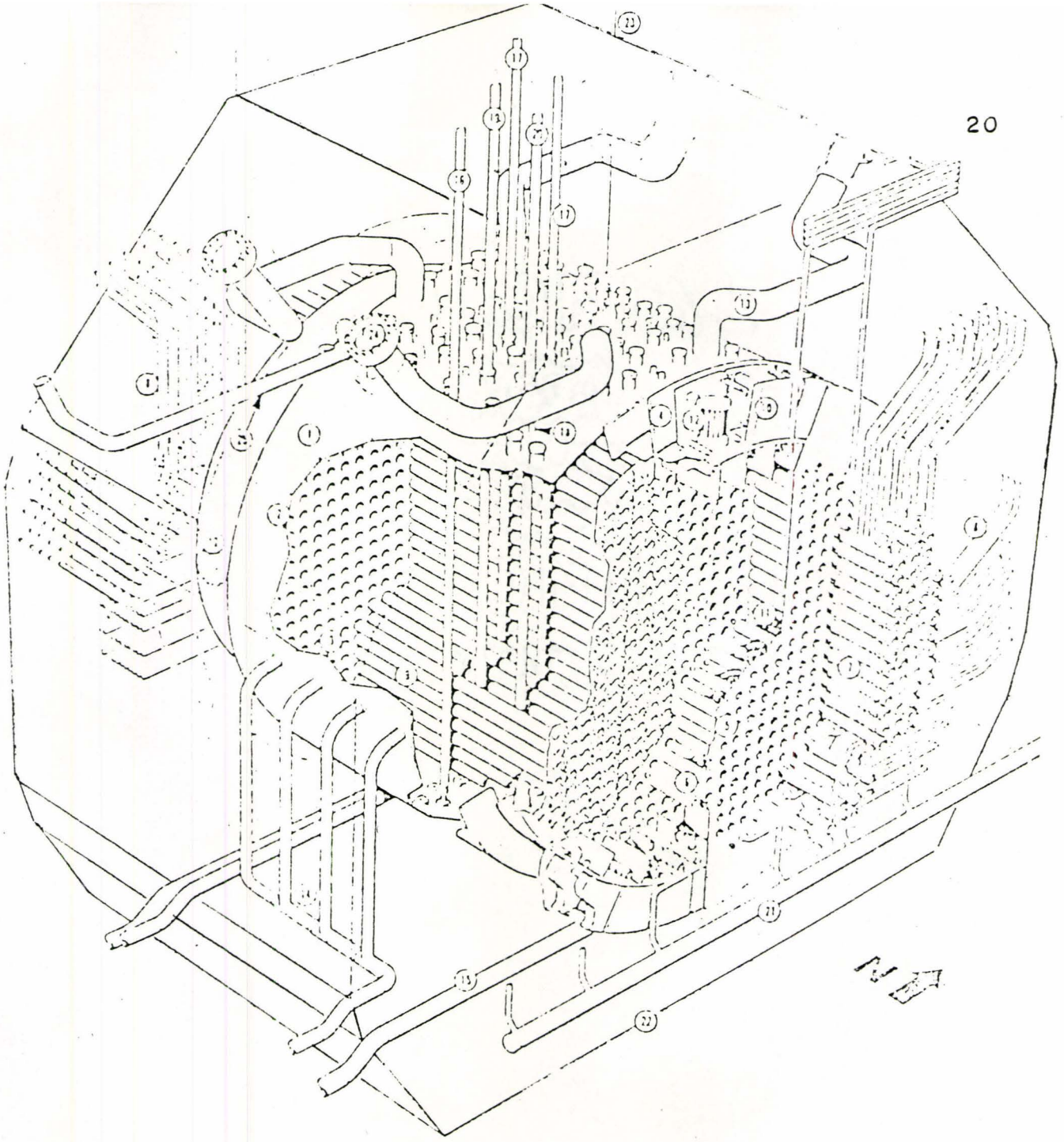
iv) If it is a transient calculation, the local heat disposition routine is called, which provides or changes the necessary heat sources in the area for future calculations. This step is by-passed if it is a steady state calculation.

v) The new temperature field is calculated using the energy equation 4-4.

vi) Heat exchangers model is called to give the inlet temperatures for the moderator. The

necessary temperature boundary conditions are then imposed after this calculation.

vii) The velocity, temperature and pressure fields are then used as the starting values for the next time step cycles.



1 CALANDRIA
2 CALANDRIA SHELL
3 CALANDRIA SIDE TUBE SHEET
4 BAFFLE PLATE
5 FUELLING MACHINE SIDE TUBE SHEET
6 LATTICE TUBE
7 END FITTINGS
8 REFLUX
9 SHIELD TANK BAFFLES
10 SHIELD TANK SIDE PLATE
11 STEEL BALL SHIELDING END SHIELD
12 MANHOLE
13 MODERATOR DISCHARGE PIPES
14 MODERATOR INLETS
15 MODERATOR OUTLETS
16 SHUT OFF UNIT
17 ADJUSTER UNIT
18 VERTICAL FLUX DETECTOR
19 CONTROL ABSORBER
20 LIQUID ZONE CONTROL UNIT
21 EMERGENCY POWER BOOSTER
22 SHIELD PLATE
23 SHIELD TANK EXPANSION
24 RUPTURE DISC ASSEMBLY
25 MODERATOR OVERFLOW

JAN. 1977

FIGURE 1 : TPYICAL MODERATOR SYSTEM

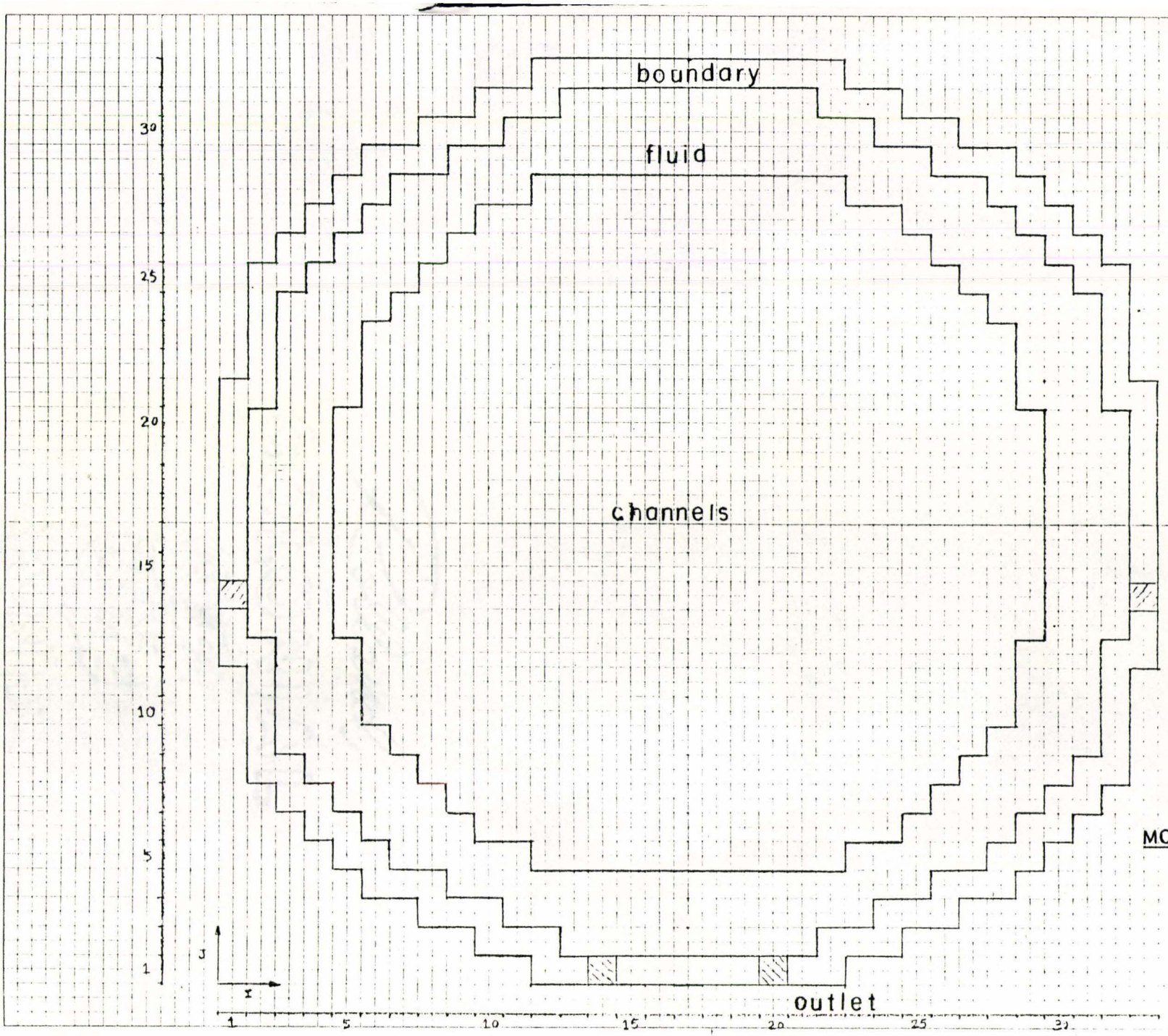


FIGURE 2
MODERATOR LAYOUT

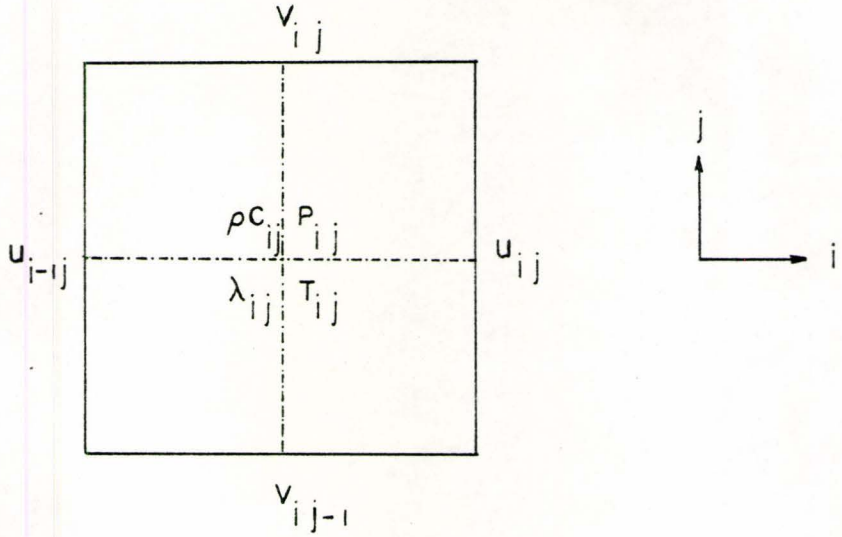


FIGURE : 3

CELL STRUCTURE

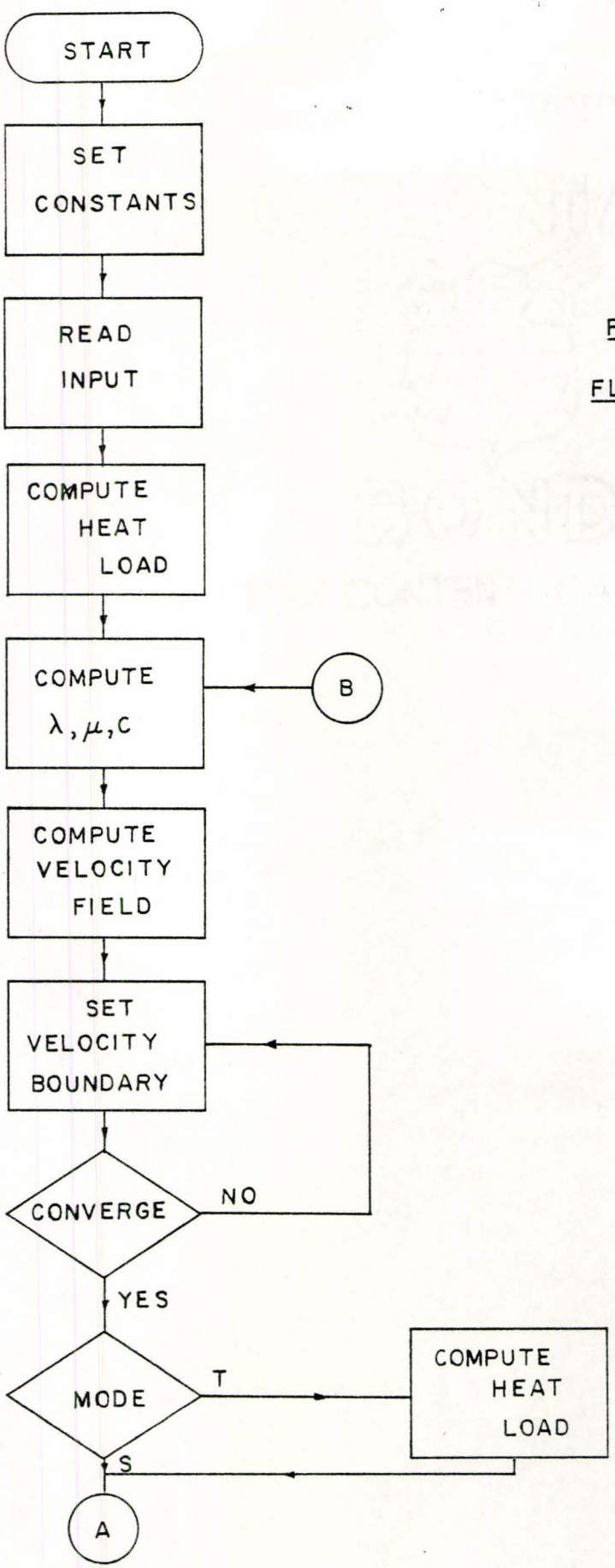
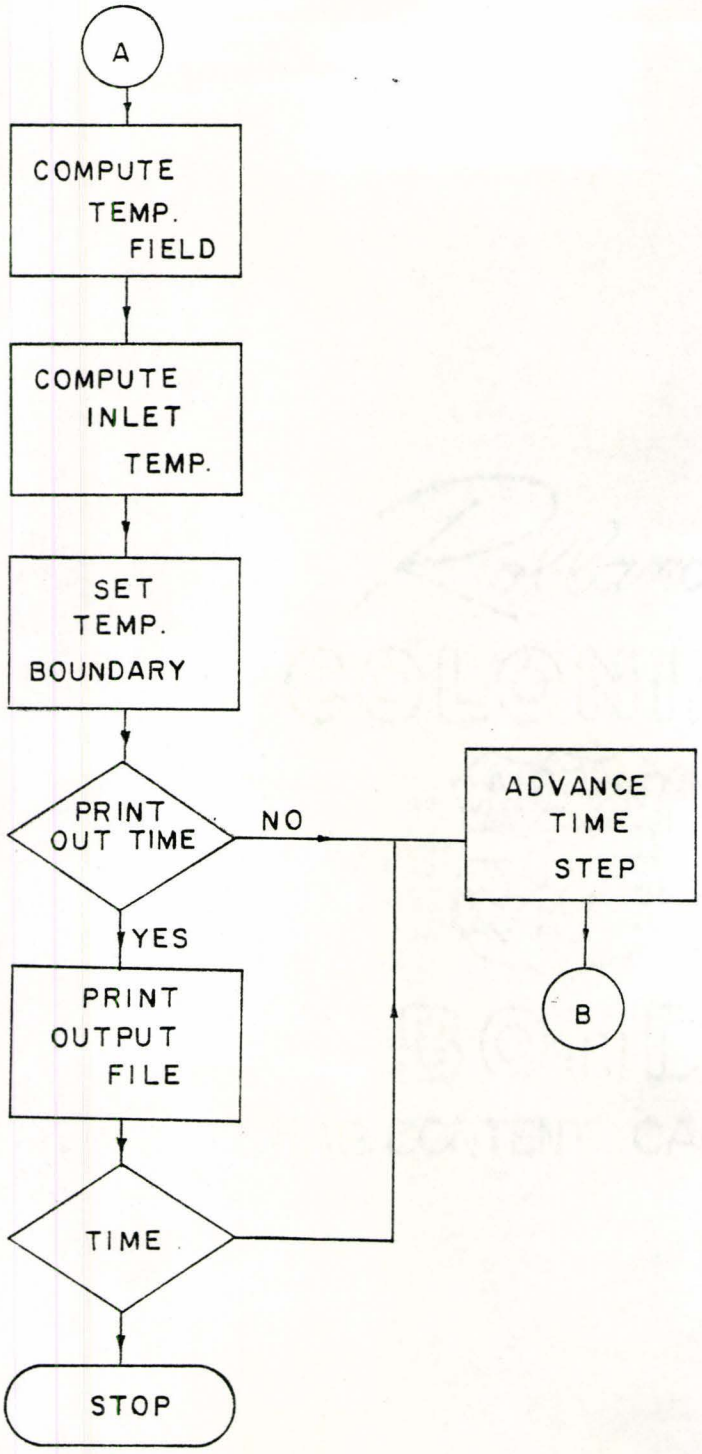


FIGURE : 4

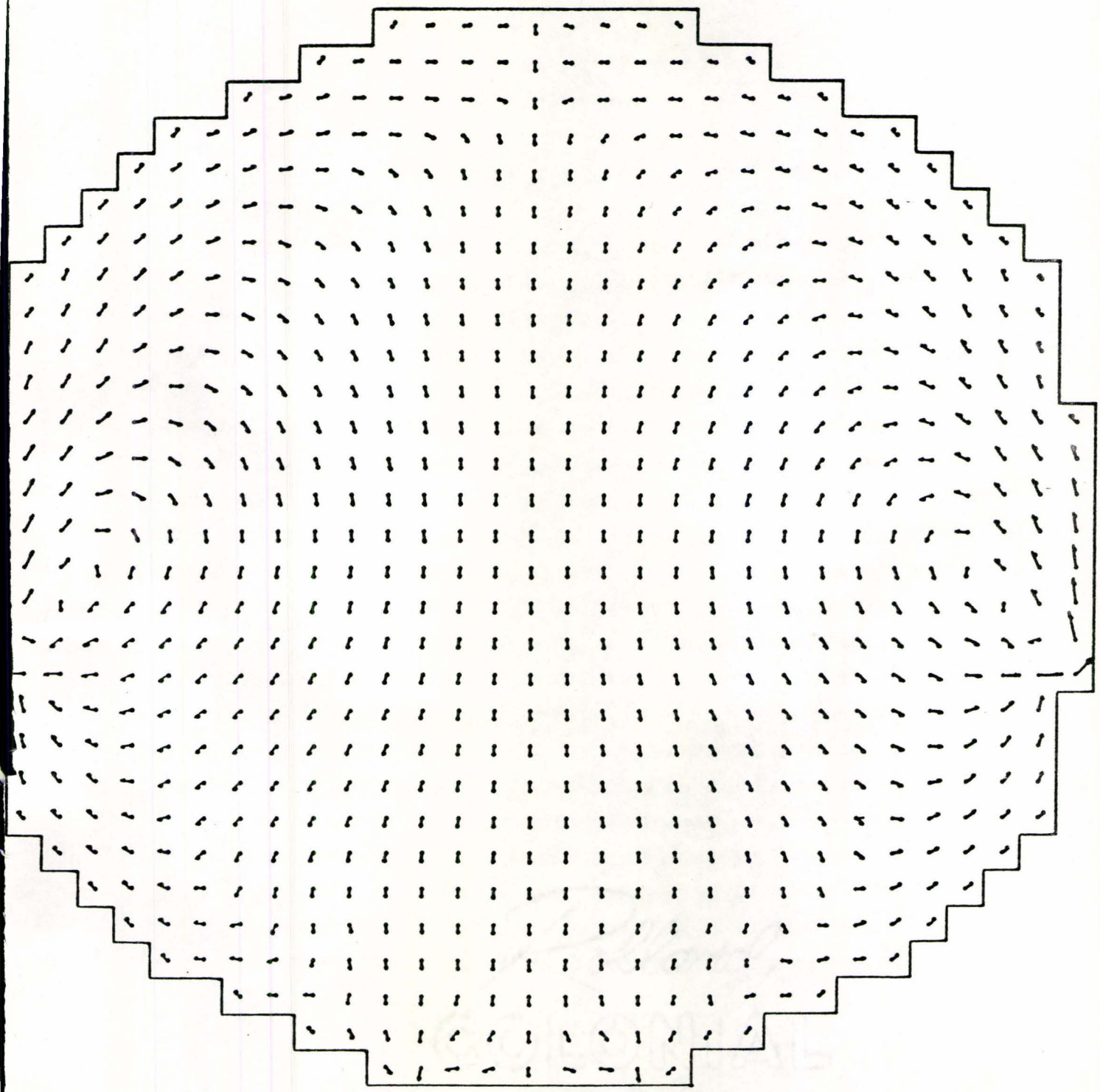
FLOW CHART


A
COMPUTE TEMP. FIELD
COMPUTE INLET TEMP.
SET TEMP. BOUNDARY
PRINT OUT TIME
NO
ADVANCE TIME STEP
B
YES
PRINT OUTPUT FILE
TIME
STOP

VII EXAMPLE

The moderator system in Darlington is used as an example. The calculations are performed without the heat exchangers model, since some of the information for the heat exchangers are not available. The nominal inlet volume flow rate is 0.89 m^3/sec at 41^{o}C. All other parameters are indicated in the assign and initial statements at the beginning of the code. Results are shown from the following figures.

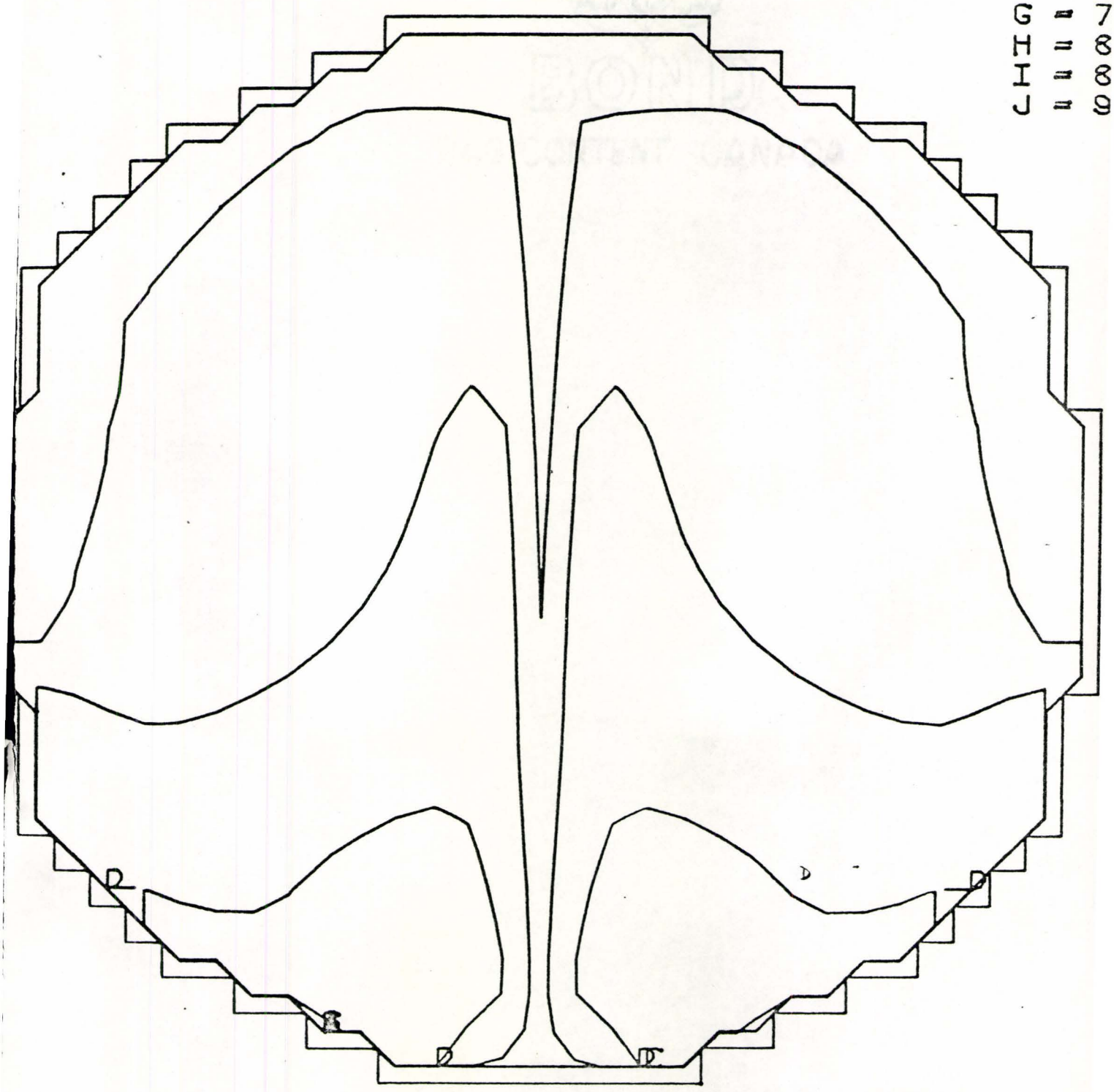
TIME = 161.4 SEC



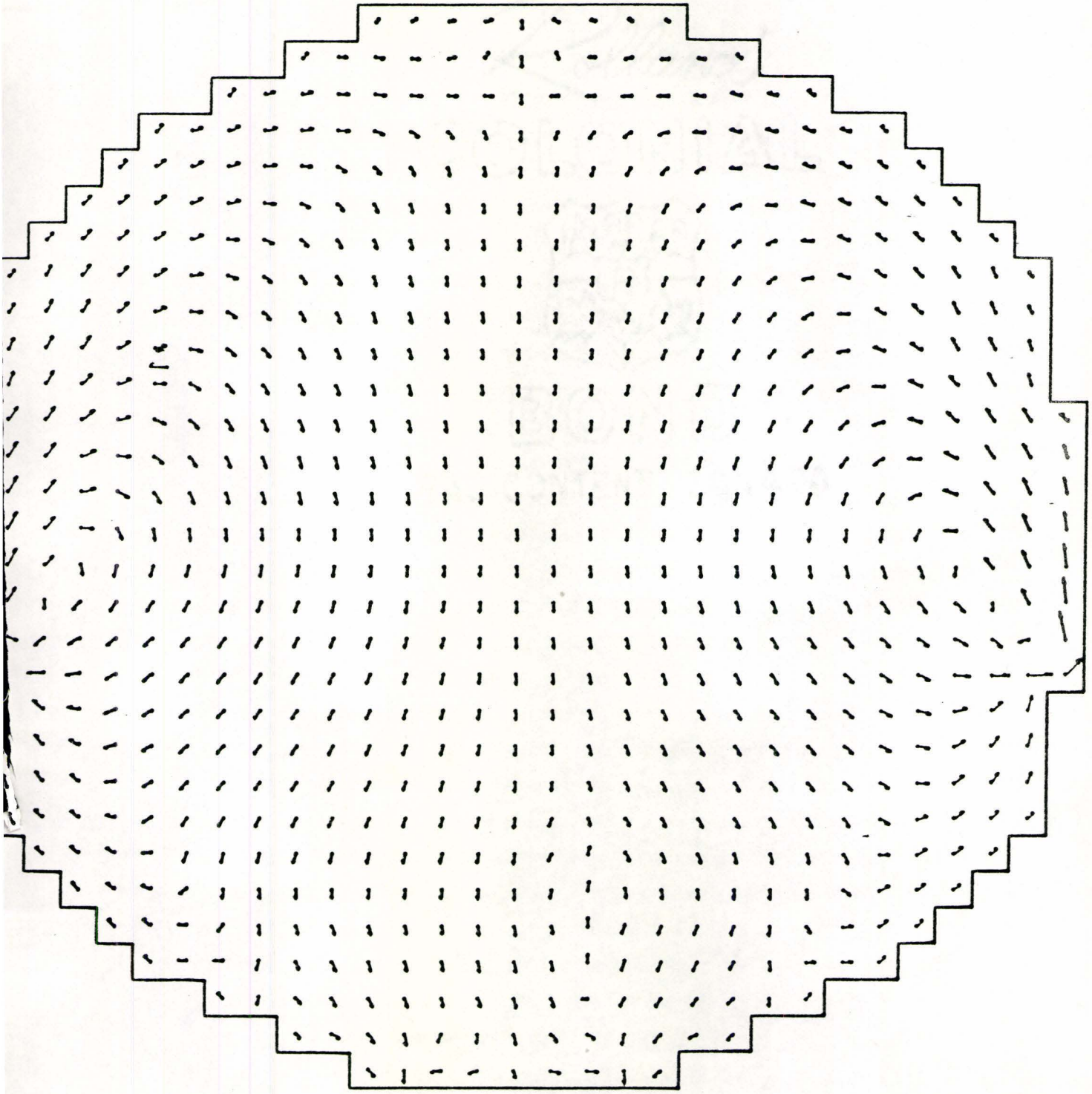
SCALE

TEMP. CONTOURS

A = 4
B = 5
C = 5
D = 6
E = 6
F = 7
G = 7
H = 8
I = 8
J = 9



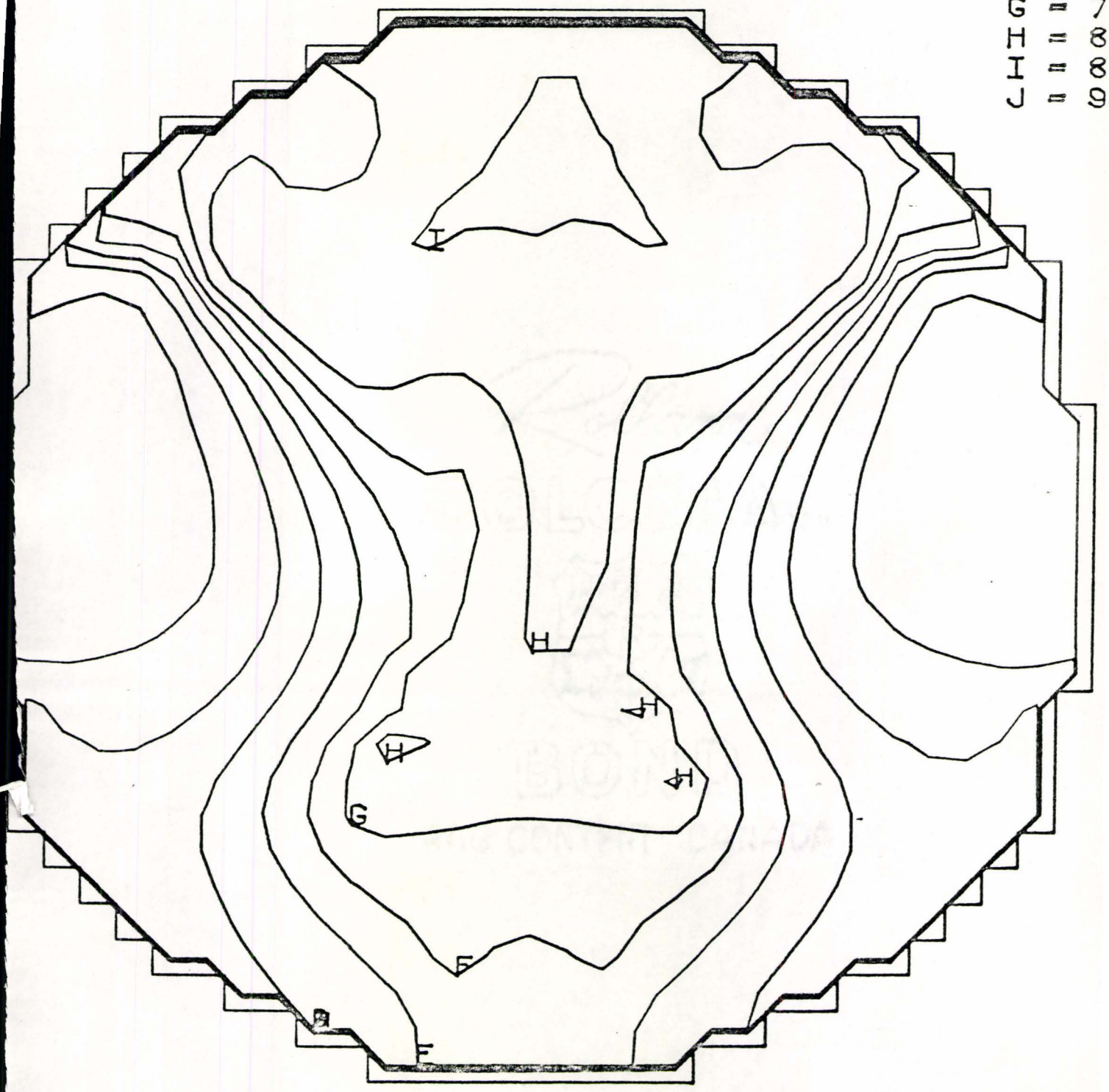
IME = 543.5 SEC



SCALE =

TEMP. CONTOURS

A = 4
B = 5
C = 5
D = 6
E = 6
F = 7
G = 7
H = 8
I = 8
J = 9



VIII CONCLUSIONS

A basic computer code to investigate the local velocities and temperatures inside the moderator has been developed.

At the present time, the code describes a 2-dimensional system, but it can be modified to 3-dimensional model with only an addition of zth component to each of the finite difference equations. 3-dimensional model can describe the effect of the non-uniformity of the axial heat flux and unevenly spaced injection of flow in the zth direction. Consequently, the velocity and temperature field will be different in each plane.

Coefficients of the Ergun equation might need to be adjusted to give a better approximation solution, because the equation applys to homogeneous porous bed with many particles. This can be achieved by either reduce the drag force term or replace it by an expression for single cylinder. Eddy viscosity is suggested to use in the viscous term for the pure fluid term.

References

(1) A.M. Chan, (1979) 3-D Numerical analysis of transient natural convection in rectangular enclosures, 101, 114-119.

(2) Combaruous M.A. and Bories S.A. (1975), Hydrothermal convection in saturated porous media, Advances in Hydroscience (NT) 10, 231-307.

(3) Bird, R.E. et al, transport phenomena, J. Wiley (1960).

(4) Chapman A.J. (1960). Heat transfer.

(5) Jaw L. (1964), Temperature relations in shell and tube exchangers having one pass split-flow shells, ASME transactions, series C. 86, 408-415.

(6) Keil H. (1979), MODHT: A Computer program to predict transient moderator temperatures, AECL.

(7) Aydogdu, K. (1979), Darlington NGS A: Moderator system heat load following an SDS1 trip. Memorandum to C.K. Choo, 38-03260F, 91-03208-04.

Appendix A

Documentation of MODCONV

A-1 General

The computer program MODCONV makes use of the idea of free convection in a porous medium to calculate the local velocity and temperature distributions inside the moderator. Fortran and some basic plotting routine found in CALCOMP plot library are used in the code.

The code can be roughly divided into steady state and transient calculations. The former calculation provides of the distribution of the local temperatures and velocities under normal operation of the moderator, as well as a starting point for the reactor shut down case run. A restart option is provided any time between runs. This is achieved by writing the result on a tape at the end of each run. The normal operation run is standard, since most of the information required are under these conditions. Therefore, they are initialized at the beginning of the program instead of having a long input card deck.

There are several options for the transient calculations which corresponds to a shut down case run. The inlet nozzle flow rate, both in magnitude and angle can be changed at the beginning of each continous run. This simulates a loss of class IV power in the pumps or abnormal flow condition of D_2O in the system. Since the inlet temperature is coupled by the heat exchangers, change of flow rate will result in change of inlet velocity and temperature. The local heat load is allowed to change with respect to time either in a continuous or stepwise manner. A continuous decrease of heat load with a continuous increase (with bounds, or fix valve) of heat load in some particular zones simulates a pressure tube break outside core shut down case.

A-2 Main MODCONV

Purpose:

To find the velocity and temperature distribution of the moderator.

Theory:

Set up constants, routines for the subprograms and format the printout.

Procedures:

See the calculations procedures and flow charts.

Common Blocks:

/BLOCK1/ UN, VN, TP, P

/BLOCK2/ U, V

/BLOCK3/ INDEX1, INDEX2, INDEX3, DELT, DEXX, DETT, DEZZ

/BLOCK4/ PO, VOR, CNF, RAD, TAV, TOI, BETA

/BLOCK5/ TPN, QNG

/BLOCK6/ UM, ANGLE, UREF

/BLOCK8/ HTEMP, HVISC, HCOND, NDH

/BLOCK9/ TEMPI, VROR, CNDF, DECA, DDEN, ND

/BLOCKA/ TD2ØM, TH2Ø, PSFLOW, Z2, Z3, Z5, Z6,
Z7, AX, DI, DO, AHI, PSF02, AHO, RC, RW

/BLOCKB/ ACTM, ARRY, IY, IZ, JY, JZ, TSHUT, NZ, ITAP

Supporting functions:

ATAN2 - Antan

HEAT - to produce local heat load

HTEXCH - heat exchangers model

LINT - linear interpolate routine

OUTPLOT - plot velocity and temperature field

PLOT	-	plot two points routine
PLOTID	-	open plotting routine
SECOND	-	return CP second ellpsed
SQRT	-	square root
TAN	-	tan
TEMPFD	-	compute temperature field
TRPLOT	-	plot power and temperature history
VELFD	-	compute velocity field
TAPE1	-	mass storage for velocities, temperatures and pressures
TAPE2	-	mass storage for power and temperature history in different locations
TAPE3	-	mass data stranfer for heat disposition after shut down in different zones

Nomenclatures:

AHI - tube side heat transfer coefficient under design conditions (kw/m^2C)

AHO - shell side heat transfer coefficient under design conditons (kw/m^2C)

ANG - angle of inflow (in degree)

ANGLE(I,J) - angles of velocities for each cell (in radian)

AX - total outside tube heat transfer surface area (m^2)

BETA - product of volumetric expansivity and gravitational acceleration

BETA3 - pressure fluctuation adjustment

CCP - specific heat capacity for D_2O ($kJ/kg\ C^o$)

CNDF(I) - Conductivities of D_2O with respect to TEMP1 ($kJ/ms\ ^oC$)

CNF(I,J,K) - Conductivities of D_2O at the cell temperature ($kJ/ms\ ^oC$)

CP - specific heat capacity for H_2O ($kJ/kg\ C^o$)

CYCLE - number of times of Δt calculations.

D - divergence of the velocities

DAC(I,J,K) - products of density and specific heat capacity of D_2O at the cell temperature.

DDEN(I) - densities of D_2O with respect to TEMPI (kg/m^3)

DECA(I) - products of density and specific heat capacity of $D_2^0$with respect to TEMPI

DELP - ΔP

DELT - Δt

DEXX - ΔX

DEYY - ΔY

DEZZ - ΔZ

DI - internal diameter of the tube in the heat exchangers

DO - external diameter of the tube in the heat exchangers

DROR - density of D_2O (dummy) (kg/M^3)

EPSI - convergence criterion

FLG - iteration check indicator

GY - gravitational acceleration (m/sec^2)

HCOND(I) - conductivities of H_2O with respect to HTEMP ($_{KJ}$/msi)

HTEMP(I) - temperature arrays for H_2O

HVISC(I) - viscosity of H_2O with respect to HTEMP

IA - array index for time out

IACT - accident indicator

ICON - continuous mode indicator

IFLOW - inlet flow charge indicator

IMAP - maps printing indicator

IN(I) - boundary indexing adjuster

INDEX1(I,J)- start and ends of the cell numbers for the overall core.

INDEX2(I,J)- start and ends of the cell numbers for the fluid

INDEX3(I,J)- start and ends of the cell numbers for the fuel channels

ISCALE - scale selection indicator

IT - time mode indicator

ITAP - tape type indicator

ITER - number of iterations

ITRAN - state mode indicator

NAME(I) - character strip for the title

ND - array dimension for TEMPI

NDH - array dimension for HTEMP

NPX,NPY - locations number for transient power plotting

NT - array dimension for TIME

NTP - number of locations for transient power plotting

OMG - relaxation factor

P(I,J,K) - pressure

PMOD(I) - total power in MW disposed into the channel region with respect to TIMQ

PO(I,J) - porosity

POR - porosity

POWM - total power in MW disposed into the Channel region

POWR - TOTAL POWER IN MW disposed into the fluid region

PREF - total power in MW disposed into the fluid region with respect to TIMQ

PSFLOW - mass flow rate of H_2O (Kg/sec)

PSFOZ - mass flow rate of H_2O at design condition

QFL - volume flow rate of D_2O (m^3/sec)

QNG(I,J) - local heat source per unit volume (kw/m^3)

QST - instanteous total heat load (kw)

QVO - net heat outflow (kw)

RAD - radius of the fuel channels (m)

RC - thermal resistance due to fouling (m^2C/kw)

RW - thermal resistance of the tube material (m^2C/kw)

T - real time (sec)

TAV - volume average temperature (^oC)

TD2ØM - mean temperature of D_2O at design valve

TEMPI(I) - temperature array for D_2O properties

TFLO - average temperature of the moderator outflow

THZI - inlet temperature of the H_2O

THZØ - outlet temperature of the H_2O

TH2ØM - mean temperature of the H_2O at design value

TI - initial temperature

TIA - accumulated time for stepwise mode timing

TIME(I) - printout times

TIMQ(I) - time array for heat load history

TO - design outlet temperature

TOI - initial reference temperature

TP(I,J,K) - temperature field at nth time step

TPN(I,J,K) - temperature field at n+1th time step

TSHUT - shut down time

U(I,J,K) - xth component of the velocity field at nth time step

UBYI - xth component of the inlet velocity

UM(I,J) - magnitudes of the velocities

UN(I,J,K) - xth component of the velocity field at n+1th time step

UREF - magnitude of the maximum velocity

UU(I) - xth components of the velocities at cell center

V(I,J,K) - yth components of the velocity field at nth time step

VBYI - yth components of the inlet velocity

YBYO - yth components of the outlet velocity

VN(I,J,K) - yth components of the velocities field at n+1th time step

VOL - volumetric expansivity of D_2O

VOR(I,J,K) - viscosities of D_2O at each cell divided by density at TAV

VROR(I) - visccosities of D_2O with respect to TEMPI

VV - yth components of the velocities at cell center

WF - wall type indicator

ZWN - computer time consumed

Z1 - viscosity of D_2O at design temperature

Z2 - conductivity of D_2O at design temperature

Z3 - Prandtl number of $D_{2Z}O$ at design temperature

Z5 - viscosity of H_2O at design temperature

Z6 - conductivity of H_2O at design temperature

Z7 - Prandtl number of H_2O at design temperature

A3 SUBROUTINE HEAT (QNG, POWR, POWM, IMOD)

Purpose: To assign the local heat dispositions for array QNG

Theory = The calculation is divided into two parts:

1. normal heat disposition

2. heat disposition with arbitrary heat disposition in some area

1. The assignment for QNG divides into 2 regions

a Fluid region where uniform heat flux is assumed

QNG(I,J) = total power in fluid region (KW)/no. of cell in the fluid region

b Fuel channels region where power channel flux distribution is assumed.

QNG(I,J) =

where N is the total number of cells in the fuel channels region and $h_{I,J}$ is the normalized distribution factor,

2. This option is selected if IMOD = 0

All the calculations in part one are performed, but in addition, the desired heat dispostion historys in some particular zones are inputed from TAPE3 and overlayed on QNG(I,J). Power inputs are fuel channel heat dispositions in KW per cell. TAPE3 format is

$TIME_1$, $POWER1_1$, $POWER2_1$, ...$POWERN_1$, $TIME_2$, $POWER1_2$, ...$POWERN_2$

...where the superscribe is for the time step and the number behind the power indicates the zones.

Procedures : overlay method is used to assign the array QNG

Variables input : POWR, POWM, IMOD, TAPE3

COMMON BLOCKS : BLOCK3, BLOCKB, #

Variables changed : ONG(I,J)

Supporting functions : EOF - end of file indicator

Options : IMOD=0 : power input is obtained through FP

IMOD=1 : power input is obtained through FP plus TAPE3 in certain zones.

Nomenclatures:

ACTM(I)	-	time in nth x n+1th interval
ARRY(I,J)	-	power in kw/cell for the fuel channel
DUM(I)	-	interpolated power
F	-	$h_{I,J}$
FRACT	-	slope for the interpolation
IMOD	-	mode indicator
IY(I)	-	initial cell number in x-direction for zone I
IZ(I)	-	final cell number in x-direction for zone I
JY(I)	-	initial cell number in y-direction for zone I
JZ(I)	-	final cell number in y-direction for zone I
NZ	-	zone number
PMFCT	-	power factor, used in fuel channel region
POWM	-	total power input per volume for fuel channel region (MW)
POWR	-	total power input per volume for fluid region (MW)
PRT01(I)	-	normalized distribution factor, hI,J
PRT02(I)	-	normalized distribution factor in the centre filling
QNG(I,J)	-	local heat source per volume (kw/m^3)

A-4 SUBROUTINE HTEXCH (T2, TI, T3, QFA, TH2I)

Purpose : To calculate the outflow temperature of D_2O and H_2O in the heat exchangers using NTU Method

Theory : refer to Appendix B

Procedures : direct computations

Variables input : T2, TI, QFA, TH2I, BLOCK8, BLOCK9, BLOCKA

Common blocks : BLOCK8, BLOCK9, BLOCKA

Variables changed : TI, T3, TH2Ø

Supporting functions : ALOG - log
EXP - e
LINT - linear interploate routine

Nomenclatures:

CR	-	ratio of the mass flow rate of D_2O to the mass flow rate of H_2O
C1	-	mass flow rate of D_2O
C2	-	mass flow rate of H_2O

EEF	-	thermal effectiveness * CR
HI	-	tube side heat transfer coefficient
HO	-	shell side heat transfer coefficient
P	-	thermal effectiveness
PHI	-	exp (UA/WC)
PRN	-	Prandt number
QFA	-	QFL - 0.036, 0.036 is the volume flow rate through the purification system

Note: variable names not listed have the same meaning as in the main program.

A-5 SUBROUTINE LINT CPT, T, PI, TI, N)

Purpose : to interpolate (PI,TI) from et of points (PT,T)

Theory :

Procedures :

1. Scan points to check if (PI, TI) is within range. If not, a warning message is printed.

2. values of P is then calculated using the above equation.

Variables input : PT, T, TI, N

Common Blocks : Nil

Variables changed : PI

Supporting function : Nil

Nomenclature:

N - dimension of array PT and T

PI	-	point want to find, with TI known
PT	-	Array for PI
T	-	Array for TI
TI	-	point input for interpolation

A-6 SUBROUTINE OUTPLOT (TPRT1, TPRT2, ITRAN, ISC)

Purpose : To produce velocity and temperature plots

Theory : 1. Linear graph

2. log scaled graph

3. No scale graph

where $U_{i,j}$ is the magnitude of the velocity at (i,j) on the plot

$U_{i,j}$ is the magnitude of the velocity at (i,j)

U_{max} is the magnitude of the maximum velocity

Procedure :

1. The moderator boundary is traced
2. The necessary heading is printed
3. Velocities, together with the direction, arrows are plotted

4. Call PLTCON to produce the temperature plot

Variables input : TPRT1, TPRT2, ITRAN, ISC

Common blocks : BLOCK3, BLOCK6, BLOCK5

Variables changed : Nil

Supporting functions : ALOG - log
COS - cos
NUMBER - to print real numbers from CALCOMP plotting library
PLOT - to plot two points from CALCOMP plotting library, 3 arguments
SIN - sin
SYMBOL - to draw characters from CALCOMP plotting library
PLTCON - to produce contours

Options : ITRAN=0 TPRT1 is printed as time (real time)
ITRAN=1 TPRT2 is printed as time (shut down time)
ISC=0 linear scaled
ISC=1 log scaled
ISC=2 no scale

Nomenclature:

ADJ	-	adjustment of the contours
ANGLE(I,J)	-	angles of the velocities in radians
D(I,J)	-	temperature arrays for contours plot
FV(I)	-	values of temperature in the contours
ISC	-	scale indicator
SIZE	-	scaling factor for plots (plotter unit/data unit
TIP1X, TIP2X	-	x-coordinates for the arrow head
TIP1Y, TIP2Y	-	y-cordinates for the arrow head
TPRT1	-	real time (in second)
TPRT2	-	shut down time (in second)
UM(I,J)	-	magnitudes of the velocities
UMT	-	magnitudes of the velocity after proper scaling
UNOR	-	normalized velocity in linear scale plot
UX	-	scaled velocity with respect to 0.2
UREF	-	magnitude of maximum velocity

A-7 SUBROUTINE TEMPFD

Purpose : To compute the temperature field

Theory : Refer to the finite differencing energy equation 4-4

Variables input : through common blocks

Common blocks : BLOCK1, BLOCK3, BLOCK4, BLOCK5

Variables changed : TPN (I,J,K), TAV

Supporting functions : ABS

Nomenclature:

CNF(I,J,K)- conductivity of the D_2O in (i,J,K) at t^n

DAC(I,J,K)- product of density and specific heat capacity of the D_2O in (i,j,k) at t^n

QNG(I,J,K)- local heat density (KW/m^3)

PO(I,J) - porosity

T(I,J,K) - temperature field in (i,j,k) at t^n

TAV - volume averaged temperature

TPN(I,J,K)- temperature field in (i,j,k) at t^{n+1}

NOTE : If the number of cells is changed, statement TAV=TAV/750 need to be adjusted, since 750 is the total number of cells for the system.

A-8 SUBROUTINE TRPLOT (TSHUT, NTP, DEXX, DEYY, DEZZ)

Purpose : To recall and plot the temperature and local heat transient in some particular locations

Procedures :

1. Read temperature and local heat from tape 2
2. Use standard plotting technique to print and plot the temperature and local heat history

Variables input : TSHUT, NTP, DEXX, DEYY, DEZZ, TAPE2

Common blocks : Nil

Supporting functions :

AXISN -	to plot axis from CALCOMP plotting library
EOF -	end of file
PLOT -	to plot two points from CALCOMP plotting library
SYMBOL -	to draw characters from CALCOMP plotting library

Nomenclature:

KVAL(I)	-	character string which contains the location identifier
NQP=NTP	-	no. of location points
QEP(I)	-	array for local heat sources in a particular time
QEPI(I)	-	array for initial local heat sources
QMAX	-	maximum local heat source
SIZE	-	scaling factor for heat source (plotter unit/ data unit)
SIZEQ	-	scaling factor for heat source * volume
SIZEX	-	scaling factor for time (plotter unit/data unit)
SIZEY	-	scaling factor for temperature (plotter unit/data unit)
TEI	-	initial time (in second)
TEP(I)	-	array for temperature in a particular time
TEPI(I)	-	array for initial temperature
TIME	-	corresponding time for temperature and heat
TMAX	-	maximum temperature
TSHUT	-	final time for printing
VOL	-	volume of the cell

A-9 SUBROUTINE VELFD

Purpose : To compute the velocity field

Theory : Refer to the differencing momentum eq 4-2, 4-3

Variables input : through common blocks

Common blocks : BLOCK1, BLOCK2, BLOCK3, BLOCK4

Variables changed : UN(I,J,K), VN(I,J,K)

Supporting functions : $ABS(x)^{-1x1}$

Nomenclature:

BETA	-	product of thermal expansitivity and gravitational acceleration
CNF	-	conductivity of the D_2O
COEX,COEY	-	/K with x and y direction choice of porosity
PO	-	porosity
PXØ,PX1,PX2	-	porosity at (i-0/2,j), (i+1/2,j), (i+1-1/2,j) respectively
PYØ,PY1,PY2	-	porosity at (i,j-1/2), (i,j+1/2), (i,j+1-1/2) respectively

RAD - radius of the fuel channels

TAV - volume averaged temperature which is also used as reference temperature

TP(I,J,K)- temperature field at t^n

U(I,J,K),v(I,J,K)- velocity field in x and y direction at t^n

UN(I,J,K),VN(I,J,K)-velocity field in x and y direction at t^n

VDR(I,J,K)- viscosity at (i,j,k)/density at TAV

A-10 INPUT CARDS

Card 1 (10A8)

NAME(I), I-1, 10

This is the title card. Character string of less than 80 are acceptable

Card 2 (10F8.3)

Time(I), I=1, 10

This controls the roll out time. There are three options on selecting the type of roll out and termination time, see indicator IT in next card for details. The last time input serves as the termination time. A maximum of 10 is allowed. A blank or zero according to the computer system is used to terminate the string.

Card 3 (7I5)

ICON, IT, ITRAN, IMAP, ISCALE, IACT, IFLOW

ICON - to indicate a initial or restart run

ICON=0 means initial run

ICON=1 means restart run

Tape 1 should be ready if ICON=1 because data are read from this tape.

IT - to indicate the type of roll out and termination time

IT=0 means real time (in second)

IT=1 means CP time (in second)

IT=2 means stepwise increase time (in second)

note: if IT=2 is used, TIME(1). TIME(2), TIME(3) are the initial, final, and step increment time respectively. Any other values follow are neglected.

ITRAN - to indicate mode of computation

ITRAN=0 steady state calculation

ITRAN=1 transient calculation

IMAP - maps selecting parameter

IMAP=0 temperature and heat source maps

IMAP=1 temperature and heat source maps and velocity and temperature plot(s)

IMAP=2 temperature and heat source maps and velocity map(s)

IMAP=3 temperature and heat source maps and velocity and temperature plot(s) and map(s)

ISCALE - to indicate type of scale in the velocity plot(s)

IXALE=0 linear scale

ISCALE=1 log scale

ISCALE=2 no scale, only directions are shown

The following indicators are used, only if ITRAN=1

ITAP - input type indicator This parameter is used only if

ITRAN=1 and IACT=1

ITAP=3 Power transients information are read in from tape 3

ITAP=5 Power transients information are read in from cards

IACT - accident indicator

IACT=0 no accident shut down

IACT=1 with accident shut down

IFLOW - inflow change parameter

IFLOW=0 flow shut off

IFLOW=1 both flow magnitude and angle need to change

IFLOW=2 no change in flow, previous values are presumed

Card 4 (4F10.4)

TAV, QFL, ANG, DELT

This card is omitted if ICON = 1

TAV - initial reference temperature (^{o}C) QFL - inlet volume flow rate (m^3/sec) ANG - angle of inlet flow respect to horizontal (^{o}) DELT - time increment (sec)

Card 5 (2F10.4)

QFL, ANG

This card is read only if IFLOW = 1 and ICON = 1

QFL - new flow magnitude (m^3/sec) ANG - new flow angle (^{o})

Cards 6, 7, 8 (16I5)

NTP, (NPX(I), NPY(I), I = 1, NTP)

This card is read only if ITRAN = 1 and ICON = 1

NTP - number of coordinates want to trace and temperature _ and local heat since time history

NPX(I) - call number in X direction NPY(I) - cell number in Y direction

Cards 9-20 (4I5)

IY(I), IZ(I), JY(I), JZ(I) = I = 1,20

These cards are read only if ICON = 1 and IACT = 1

IY, IZ, JY, JZ are the initial and final cell number for the zones which undergo a different heat disposition in X and Y direction respectively. Maximum number of zones is 20. If IY or IZ are declared outside the fuel channel region, it is truncated automatically. Values of JY and JZ should be within 5 and 28, where the fuel channel region are.

If this option is used, tape 3 or input cards deck, depending on ITAP, need to be ready for the code. One blank card is required to follow this set of cards, if less than 20 zones are used.

Card 29 (Fb.2, 4(5E15.8,1))

ACTM, ARRY

These cards are read only if ITAP = 5 and IACT = 1

ACTM - time after shut down (sec) ARRY - corresponding fuel channel powers for each zone
(kw/cell)

These should have sufficient cards to cover the desired running period, otherwise the "powers in the last time valve is used and a warning message - power is extrapolated" will be printed.

A-11 Getting the Outputs

Outputs depend on the indicators declared and type of computations.

1. In steady state calculations, the type of print out depends on the indicator IMAP. Basically, the inlet and outlet flow magnitudes and porosity map are printed no matter what type of calculations. Print out maps are in the sequence of velocities, temperatures and local heat dispositions, with the control of IMAP as described in "input card" section. Plotting tape consists of the velocities and temperatures plots in different roll out time. TAPE1 will automatically filled with velocities, temperatures and pressures for next continuous run before the end of execution.

2. In transient calculations, a history of temperatures and heat dispositions in the desired locations are printed, in addition to the plots described above. There are two plots showing the history of the temperatures and heat dispositions follow the velocities and temperatures plots for each roll out time in the plotting tape.

```
 PROGRAM MODCONV (INPUT=65,OUTPUT=65,TAPE5=INPUT,TAPE6=OUTPUT,
1TAPE1,TAPE2=65,TAPE3=65)
 DIMENSION U(33,32,1),V(33,32,1),TP(33,32,1),P(33,32,1),
1UN(33,32,1),VN(33,32,1),TPN(33,32,1),TIMQ(20),PREF(20),PMOD(20),
2PO(34,33),NAME(10),INDEX1(2,32),INDEX2(2,32),INDEX3(2,24),
3QNG(33,32),IN(2),UU(12),VV(12),UM(33,32),ANGLE(33,32),
4 TIME(10),VOR(33,32,1),CNF(33,32,1),DAC(33,32,1),NPX(15),NPY(15)
 DIMENSION HTEMP(19),HVISC(19),HCOND(19),
1TEMP1(19),VROR(19),CNDF(19),DECA(19),DDEN(19)
 DIMENSION ACTM(2),ARRY(2,20),IY(20),IZ(20),JY(20),JZ(20)
 INTEGER CYCLE
 COMMON /BLOCK1/ UN,VN,TP,P
 COMMON /BLOCK2/ U,V
 COMMON /BLOCK3/ INDEX1,INDEX2,INDEX3,DELT,DEXX,DEYY,DEZZ
 COMMON /BLOCK4/ PO,VOR,CNF,DAC,RAD,TAV,TOI,BETA
 COMMON /BLOCK5/ TPN,QNG
 COMMON /BLOCK6/ UM,ANGLE,UREF
 COMMON /BLOCK8/ HTEMP,HVISC,HCOND,NDH
 COMMON /BLOCK9/ TEMP1,VROR,CNDF,DECA,DDEN,ND
 COMMON /BLOCKA/ TD20M,TH20,PSFLOW  ,Z2,Z3,Z5,Z6,Z7,
1AX,DI,DO,AHI,PSFO2,AHO,RC,RW
 COMMON /BLOCKB/ ACTM,ARRY,IY,IZ,JY,JZ,TSHUT,NZ,ITAP
 DATA PO/1122*1./
 DATA ((INDEX1(I,J),I=1,2),J=1,20)/12,22,10,24,8,26,6,28,5,29,4,30,
13,31,4*(2,32),9*(1,33)/
 DATA ((INDEX2(I,J),I=1,2),J=1,20)/0,0,13,21,11,23,9,25,7,27,
16,28,5,29,4,30,4*(3,31),8*(2,32)/
 DATA ((INDEX3(I,J),I=1,2),J=1,12)/12,22,10,24,9,25,8,26,
17,27,3*(6,28),4*(5,29)/
 DATA IN/1,-1/
 DATA HTEMP/0.,10.,20.,25.,30.,35.,40.,45.,50.,55.,60.,65.,70.,75.,
180.,85.,90.,95.,100./
 DATA HVISC/1794.E-6,1405.E-6,1054.E-6,889.E-6,802.E-6,723.E-6,
1641.E-6,599.E-6,542.E-6,496.E-6,467.E-6,418.E-6,397.E-6,380.E-6,
2368.E-6,351.E-6,331.E-6,310.E-6,298.E-6/
```

```
      DATA HCOND/.5484E-3,.5726E-3,.5969E-3,.6072E-3,.6176E-3,.6245E-3,
     1.6332E-3,.6401E-3,.6470E-3,.6522E-3,.6591E-3,.6617E-3,.6712E-3,
     2.6758E-3,.6747E-3,.6764E-3,.6770E-3,:6773E-3,.6788E-3/
      DATA DDEN/1104.97,1103.75,1102.54,1101.32,1100.11,1097.69,1096.49,
     11094.09,1091.70,1088.14,1085.78,1082.25,1078.75,1075.27,1070.66,
     21066.10,1061.58,1057.08,1052.63/
      DATA VROR/1253.7E-6,1119.2E-6,997.4E-6,892.3E-6,802.7E-6,726.4E-6,
     1661.3E-6,605.3E-6,556.9E-6,514.8E-6,447.9E-6,445.4E-6,416.6E-6,
     2391.0E-6,368.1E-6,347.4E-6,328.8E-6,312.0E-6,296.6E-6/
      DATA TEMP1/20.,25.,30.,35.,40.,45.,50.,55.,60.,65.,70.,75.,
     1 80.,85.,90.,95.,100.,105.,110./
      DATA CNDF/0.5817E-3,0.5893E-3,0.5964E-3,0.6029E-3,0.6088E-3,
     1 0.6142E-3,0.6192E-3,0.6236E-3,0.6276E-3,0.6311E-3,0.6342E-3,
     2 0.6369E-3,0.6391E-3,0.6410E-3,0.6424E-3,0.6435E-3,0.6443E-3,
     3 0.6447E-3,0.6448E-3/
      DATA DECA/4664.08,4651.20,4639.49,4628.8,4618.26,4603.71,4594.29,
     1 4579.86,4566.58,4547.34,4534.22,4516.23, 4498.39,4480.65,
     2 4458.23,4437.11,4415.11,4395.34,4375.78/
      DATA TIMQ/0.,0.4,0.7,1.0,1.3,1.5,1.6,1.8,2.0,2.5,4.,5.,6.6,12.,
     130.,60.,120.,300.,600.,1000./
      DATA PREF/9.3,9.03,8.03,6.16,4.52,3.49,3.15,2.97,2.91,2.83,2.73,
     12.68,2.6,2.435,2.26,2.165,2.061,1.972,1.954,1.887/
      DATA PMOD/113.64,109.655,94.58,70.03,47.402,33.293,28.549,25.971,
     125.205,24.116,22.592,21.823,20.711,18.171,15.547,14.106,12.472,
     211.057,10.452,9.664/
      DO 10 I = 1,2
      DO 10 J = 1,12
      INDEX1(I,33-J) = INDEX1(I,J)
      INDEX2(I,33-J) = INDEX2(I,J)
      INDEX3(I,25-J) = INDEX3(I,J)
10    CONTINUE
      TSHUT = 0.          $K = 1              $TH2I = 18.33.
      DEXX = 0.27384375   $DEYY = 0.28575     $DEZZ = 6.77
      VOL = 5.837E-4      $GY = 9.81          $RAD = 0.214
      OMG = 1.85          $WF = -1.           $POR = 0.558
```

```
      NDH = 19            $EPSI = 5.E-4       $IA = 1
      TI = 41.                STO = 80.           $ND = 19
      AX = 3045.39            $DI = 0.01          $DO = 0.01
      AHI = 14.42779          $PSFO2 = 2081.0     $AHO = 4.19581
      PSFLOW = 2081.          $RC = 0.17612       $RW = 0.08805

!  INCREASE POWER BY 22 PERCENT
      DO 4 I = 1,20
      PREF(I) = PREF(I)*1.22
      PMOD(I) = PMOD(I)*1.22
 4    CONTINUE

! CALCULATE CONSTANTS FOR THE HEAT EXCHANGERS
      TD20M = (TI + TO)/2.
      CALL LINT(DDEN,TEMP1,DROR,TD20M,ND)
      TH20 = DROR*(QFL-0.036)/PSFLOW*(TO-TI) + TH2I
      TH20M = (TH2I + TH20)/2.
      CALL LINT(VROR,TEMP1,Z1   ,TD20M,ND)
      CALL LINT(CNDF,TEMP1,Z2   ,TD20M,ND)
      CALL LINT(DECA,TEMP1,CCP  ,TD20M,ND)
      CCP = CCP/DROR
      Z3 = CCP*Z1/Z2
      CALL LINT(HCOND,HTEMP ,Z6,TH20M,NDH)
      CALL LINT(HVISC,HTEMP ,Z5,TH20M,NDH)
      CALL LINT(DDEN,TEMP1,DROR,TH20M,ND)
      CALL LINT(DECA,TEMP1,CP,TH20M,ND)
      CP = CP/DROR
      Z7 = CP*Z5/Z6

!  READ TITLE
      READ(5,13) NAME
      WRITE(6,9) NAME
!  READ TERMINATION TIME
      READ(5,18) (TIME(I),I = 1,10)
      DO 1 I = 1,10
```

```
      IF(TIME(I) .EQ. 0.) GO TO 2
1     CONTINUE
      NT = 11
      GO TO 3
2     NT = I
 READ INDICATORS
3     READ(5,20) ICON, IT, ITRAN, IMAP, ISCALE, ITAP, IACT, IFLOW
      IF( IMAP .EQ. 1 .OR. IMAP .EQ. 3 .OR. ITRAN .EQ. 1) CALL
     1PLOTID(0,0,0.,0.)
      IF( ICON .EQ. 1) GO TO 21
      READ(5,11) TOI,QFL,ANG,DELT
 SET INITIAL CONDITIONS
      T = 0.
      FLG=1.
      CYCLE = ITER = 0
      DO 61 J = 1,32
      IB = INDEX1(1,J)
      IE = INDEX1(2,J)
      DO 61 I = IB,IE
      UN(I,J,KD = VN(I,J,KD = P(I,J,KD = 0.
      TP(I,J,KD = TPN(I,J,KD = TOI
61    CONTINUE
      GO TO 99
21    READ(1,14)  TOI,QFL,ANG,T,DELT,CYCLE
      WRITE(6,15) T,CYCLE,DELT
      IF( IFLOW .EQ. 0) QFL = 0.
      IF( IFLOW .EQ. 1) READ(5,11) QFL,ANG
      DO 63 J = 1,32
      IB = INDEX1(1,J)
      IE = INDEX1(2,J)
      DO 63 I = IB,IE
      READ(1,17) U(I,J,KD,V(I,J,KD,TP(I,J,KD,P(I,J,KD
      UN(I,J,KD = U(I,J,KD
      VN(I,J,KD = V(I,J,KD
      TPN(I,J,KD = TP(I,J,KD
```

```
 63   CONTINUE
      IF(ITRAN .EQ. 0) GO TO 99
      READ(5,31) NTP,(NPX(I),NPY(I),I=1,NTP)
      IF(IACT .EQ. 0) GO TO 99
      DO 98 I = 1,20
      READ(5,97) IY(I),IZ(I),JY(I),JZ(I)
      IF(IY(I) .NE. 0) GO TO 98
      NZ = I
      GO TO 96
 98   CONTINUE
      NZ = 15
 96   READ(ITAP,32) ACTM(1),(ARRY(1,J),J=1,NZ)
      READ(ITAP,32) ACTM(2),(ARRY(2,J),J=1,NZ)
 99   UBYI = QFL/2./DEYY/DEZZ
      VBYI = UBYI*TAN(3.14159*ANG/180.)
      VBYO = QFL/2./DEXX/DEZZ
      BETA = VOL*GY

      POWR = PREF(1)/DEXX/DEYY/DEZZ
      POWM = PMOD(1)/DEXX/DEYY/DEZZ
      CALL HEAT(QNG,POWR,POWM,0)
      WRITE(6,19) QFL,UBYI,VBYO,VBYI

C SET POROSITY

      DO 62 J = 1,24
      IB = INDEX3(1,J)
      IE = INDEX3(2,J)
      DO 62 I = IB,IE
      PO(I,J+4) = POR
 62   CONTINUE
C  PLOT POROSITY MAP
      DO 64 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
```

```
      IB1 = 4*(IB-1)
      IE1 = IE -IB+1
      WRITE(6,16) IB1,IE1,(PO(I,J),I=IB,IE)
   64 CONTINUE
C  PRINT HEADINGS
      WRITE(6,1620)
      CALL LINT(DDEN,TEMP1,DREF,TOI,ND)
 1000 DO 60 J = 1,32
      IB = INDEX1(1,J)
      IE = INDEX1(2,J)
      DO 60 I = IB ,IE
      CALL LINT(DECA,TEMP1,DAC(I,J,KD,TPN(I,J,KD,ND)
      CALL LINT(CNDF,TEMP1,CNF(I,J,KD,TPN(I,J,KD,ND)
      CALL LINT(VROR,TEMP1,VDUM,TPN(I,J,KD,ND)
      VOR(I,J,KD = VDUM/DREF
   60 CONTINUE
      IF(CYCLE .EQ. 0) GO TO 100
      ITER = 0
      FLG = 1.0

C  COMPUTE TEMPORARY VELOCITIES

      CALL VELFD

C  SET THE BOUNDARY CONDITIONS

C  CORNER BOUNDARY
  100 DO 101 J = 2,8
      UN(INDEX2(1,J)-1,J,KD = 0.
      VN(INDEX2(1,J)-1,J,KD = 0.
      UN(INDEX2(2,J),J,KD = 0.
      VN(INDEX2(2,J)+1,J,KD = 0.
  101 CONTINUE
      J = 12
      UN(INDEX2(1,J)-1,J,KD = 0.
```

```
      VN(INDEX2(1,J)-1,J,KD = 0.
      UN(INDEX2(2,J),J,KD = 0.
      VN(INDEX2(2,J)+1,J,KD = 0.
      DO 102 J = 24,31
      UN(INDEX2(1,J)-1,J,KD = 0.
      VN(INDEX2(1,J),J,KD = 0.
      UN(INDEX2(2,J),J,KD = 0.
      VN(INDEX2(2,J),J,KD = 0.
  102 CONTINUE
      J = 20
      UN(INDEX2(1,J)-1,J,KD = 0.
      VN(INDEX2(1,J),J,KD = 0.
      UN(INDEX2(2,J),J,KD = 0.
      VN(INDEX2(2,J),J,KD = 0.
C  LEFT AND RIGHT WALLS
      DO 103 J = 9,23
      IF(J.EQ.12 .OR. J.EQ.20) GO TO 103
      UN(INDEX2(1,J)-1,J,KD = 0.
      VN(INDEX2(1,J)-1,J,KD = WF*VN(INDEX2(1,J),J,KD
      UN(INDEX2(2,J),J,KD = 0.
      VN(INDEX2(2,J)+1,J,KD = WF*VN(INDEX2(2,J),J,KD
  103 CONTINUE
C  TOP AND BOTTOM WALLS
      DO 104 J = 2,5
      UN(INDEX2(1,J),J-1,KD = WF*UN(INDEX2(1,J),J,KD
      VN(INDEX2(1,J),J-1,KD = 0.
      UN(INDEX2(2,J)-1,J-1,KD = WF*UN(INDEX2(2,J)-1,J,KD
      VN(INDEX2(2,J),J-1,KD = 0.
  104 CONTINUE
      DO 105 I = 14,20
      UN(I,1,KD = WF*UN(I,2,KD
      VN(I,1,KD = 0.
      UN(I,32,KD = WF*UN(I,31,KD
      VN(I,31,KD = 0.
  105 CONTINUE
```

```
      DO 106 J = 28,31
      UN(INDEX2(1,J),J+1,KD=WF*UN(INDEX2(1,J),J,KD
      VN(INDEX2(1,J)+1,J,KD = 0.
      UN(INDEX2(2,J)-1,J+1,KD = WF*UN(INDEX2(2,J)-1,J,KD
      VN(INDEX2(2,J)-1,J,KD = 0.
  106 CONTINUE

      IF(QFL .EQ. 0.) GO TO 107
C  FIT VELOCITY BOUNDARYS
      UN(1,14,KD = UBYI
      VN(1,14,KD = VN(2,14,KD = VBYI
      VN(1,13,KD = VN(2,13,KD = VBYI
      UN(32,14,KD = -UBYI
      VN(33,14,KD = VN(32,14,KD = VBYI
      VN(33,13,KD = VN(32,13,KD = VBYI
      VN(14,1,KD = -VBYO
      VN(20,1,KD = -VBYO

C **  HAS CONVERGE BEEN REACHED

  107 IF(FLG.EQ.0.) GO TO 2000
      ITER = ITER + 1
      IF(ITER .LT. 200) GO TO 1500
      WRITE(6,1001) CYCLE
      GO TO 5000
 1500 FLG = 0.

C  COMPUTE UPDATED CELL PRESSURE AND VELOCITIES

      DO 1600 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 1600 I = IB,IE
      PRY = PO(I,J)
      PXX = 1./(PO(I+1,J)+PRY) + 1./(PO(I-1,J)+PRY)
```

```
      PYY = 1./(PO(I,J+1)+PRY) + 1./(PO(I,J-1) + PRY)
      BETA3 = OMG/(2.*DELT*(PXX/DEXX**2 + PYY/DEYY**2))
      PU = 2.*(UN(I,J,K)/(PO(I+1,J)+PRY) - UN(I-1,J,K)/(PRY +
     1 PO(I-1,J)))/DEXX
      PV = 2.*(VN(I,J,K)/(PO(I,J+1)+PRY) - VN(I,J-1,K)/(PRY +
     1 PO(I,J-1)))/DEYY
      D = PU + PV
      IF (ABS(D) .GT. EPSI) FLG = 1.
      DELP = -BETA3*D
      P(I,J,K) = P(I,J,K) + DELP
      UN(I,J,K) = UN(I,J,K) + DELT*DELP/DEXX
      UN(I-1,J,K) = UN(I-1,J,K) - DELT*DELP/DEXX
      VN(I,J,K) = VN(I,J,K) + DELT*DELP/DEYY
      VN(I,J-1,K) = VN(I,J-1,K) - DELT*DELP/DEYY
1600  CONTINUE
      GO TO 100

C  COMPUTE TRANSIENT INTERNAL HEAT GENERATION  Q / VOL.
2000  IF(ITRAN .EQ. 0) GO TO 2001
      TSHUT = TSHUT + DELT
      CALL LINT(PREF,TIMQ,POWR,TSHUT,20)
      CALL LINT(PMOD,TIMQ,POWM,TSHUT,20)
      POWR = POWR/DEXX/DEYY/DEZZ
      POWM = POWM/DEXX/DEYY/DEZZ
      CALL HEAT(QNG,POWR,POWM,IACT)

C  COMPUTE TEMP. FIELD
2001  CALL TEMPFD

      IF(ITRAN .EQ. 0) GO TO 200
      WRITE(2,30) TSHUT
      WRITE(2,30) (TPN(NPX(I),NPY(I),K),I=1,NTP)
      WRITE(2,30) (QNG(NPX(I),NPY(I)),I=1,NTP)
 200  TFLO = (TPN(14,2,1) + TPN(20,2,1))/2.
      IF(QFL.NE.0.) CALL HTEXCH(TFLO,TI,TH20M,QFL,TH2I)
```

```
C  INSULATED WALLS
      DO 210 L = 1,2
      DO 201 J = 2,8
      T2 = (TPN(INDEX2(L,J),J,KO + TPN(INDEX2(L,J),J+1,KO +
     1TPN(INDEX2(L,J)-IN(L),J+1,KO)/3.
      TPN(INDEX2(L,J)-IN(L),J,KO = T2
  201 CONTINUE
      DO 202 JJ = 2,8
      J = 33-JJ
      T2 = (TPN(INDEX2(L,J),J,KO + TPN(INDEX2(L,J),J-1,KO +
     1TPN(INDEX2(L,J)-IN(L),J-1,KO)/3.
      TPN(INDEX2(L,J)-IN(L),J,KO = T2
  202 CONTINUE
      DO 203 J = 9,24
      TPN(INDEX2(L,J)-IN(L),J,KO = TPN(INDEX2(L,J),J,KO
  203 CONTINUE
      J = 12
      T2 = (TPN(INDEX2(L,J),J,KO + TPN(INDEX2(L,J),J+1,KO +
     1TPN(INDEX2(L,J)-IN(L),J+1,KO)/3.
      TPN(INDEX2(L,J)-IN(L),J,KO = T2
      J = 21
      T2 = (TPN(INDEX2(L,J),J,KO + TPN(INDEX2(L,J),J-1,KO +
     1TPN(INDEX2(L,J)-IN(L),J-1,KO)/3.
      TPN(INDEX2(L,J)-IN(L),J,KO = T2
      TPN(INDEX2(L,14)-IN(L),14,KO = 2.*TI-TPN(INDEX2(L,14),14,KO
      IF(QFL.EQ.0.) TPN(INDEX2(L,14)-IN(L),14,KO=TPN(INDEX2(L,14),14,KO
      DO 204 J = 2,5
      TPN(INDEX2(L,J),J-1,KO = TPN(INDEX2(L,J),J,KO
      I = 33-J
      TPN(INDEX2(L,I),I+1,KO = TPN(INDEX2(L,I),I,KO
  204 CONTINUE
  210 CONTINUE
      IB = INDEX2(1,2)+1
      IE = INDEX2(2,2)-1
```

```
      DO 205 I = IB,IE
      TPN(I,1,KO) = TPN(I,2,KO)
      TPN(I,32,KO) = TPN(I,31,KO)
  205 CONTINUE

      TI = (TPN(INDEX2(1,14),14,KO) + TPN(INDEX2(1,14)+1,14,KO))/2.
      CALL LINT(DECA,TEMP1,DUM,TI,ND)
      QVO = DEXX*DEZZ*(DAC(14,1,KO)*TPN(14,1,KO)*ABS(VN(14,1,KO)) +
     1DAC(20,1,KO)*TPN(20,1,KO)*ABS(VN(20,1,KO))-2.*DEZZ*
     2DEYY*DUM*UN(1,14,KO)*TI
      QST = (POWR+POWM)*DEXX*DEYY*DEZZ
      WRITE(6,1610) CYCLE,ITER,QST,QVO,TAV,DELT,TI,TFLO
*
C **  SET THE ADVANCE TIME VELOCITIES INTO THE U,V AND W ARRAY
      DO 1720 J = 1,32
      IB = INDEX1(1,J)
      IE = INDEX1(2,J)
      DO 1720 I = IB,IE
      U(I,J,KO) = UN(I,J,KO)
      V(I,J,KO) = VN(I,J,KO)
      TP(I,J,KO) = TPN(I,J,KO)
 1720 CONTINUE
      T = T + DELT
      CYCLE = CYCLE + 1
      IF(QVO .GE. QST) GO TO 86

      IF(IT .NE. 2) GO TO 87
      TIA = TIME(1) +IA*TIME(3)
      IF(T.GE. TIA) GO TO 86
      GO TO 1000
   87 IF(IT .EQ. 0) GO TO 85
      CALL SECOND(ZWN)
      IF(ZWN .GE. TIME(IA)) GO TO 86
      GO TO 1000
   85 IF(T .LT. TIME(IA)) GO TO 1000
```

```
C  PRINT OUTPUT

 86  WRITE(6,1630) T
     IF(ITRAN .EQ. 1) WRITE(6,1640) TSHUT
     UREF = 0.

     IF(IMAP .EQ. 0) GO TO 70
     IF(IMAP .NE. 1) WRITE(6,1650)
     DO 42 JJ = 3,30
     J = 33-JJ
     IB = INDEX2(1,J)
     IB1 = 12*(IB-1)-11
     IB2 = 13 -IB
     DO 43 I = IB,12
     UU(I) = (UN(I-1,J,K) + UN(I,J,K))/2./PO(I,J)
     VV(I) = (VN(I,J-1,K) + VN(I,J,K) )/2./PO(I,J)
     UM(I,J) = SQRT(UU(I)**2 + VV(I)**2)
     IF(UREF .LT. UM(I,J)) UREF = UM(I,J)
     ANGLE(I,J) = ATAN2(VV(I),UU(I))
 43  CONTINUE
     IF(IMAP .EQ. 1) GO TO 42
     WRITE(6,56) IB1,IB2,(UU(I),I=IB,12),IB1,IB2,(VV(I),I=IB,12)
 42  CONTINUE
     DO 44 JJ = 2,31
     J = 33-JJ
     DO 45 I = 13,21
     UU(I-12) = (UN(I-1,J,K) + UN(I,J,K))/2./PO(I,J)
     VV(I-12) = (VN(I,J-1,K) + VN(I,J,K))/2./PO(I,J)
     UM(I,J) = SQRT(UU(I-12)**2 + VV(I-12)**2)
     IF(UREF .LT. UM(I,J)) UREF = UM(I,J)
     ANGLE(I,J) = ATAN2(VV(I-12),UU(I-12))
 45  CONTINUE
     IF(IMAP .EQ. 1) GO TO 44
     WRITE(6,57) (UU(I),I=1 , 9),(VV(I),I=1 , 9)
```

```
44  CONTINUE
    DO 46 JJ = 3,30
    J = 33-JJ
    IE = INDEX2(2,J)
    IE1 = IE-21
    DO 47 I = 22,IE
    UU(I-21) = (UN(I-1,J,K) + UN(I,J,K))/2./PO(I,J)
    VV(I-21) = (VN(I,J-1,K) + VN(I,J,K))/2./PO(I,J)
    UM(I,J) = SQRT(UU(I-21)**2 + VV(I-21)**2)
    IF(UREF .LT. UM(I,J)) UREF = UM(I,J)
    ANGLE(I,J) = ATAN2(VV(I-21),UU(I-21))
47  CONTINUE
    IF(IMAP .EQ. 1) GO TO 46
    WRITE(6,58) IE1,(UU(I),I=1,IE1),IE1,(VV(I),I=1,IE1)
46  CONTINUE
70  WRITE(6,1660)
    DO 71 JJ = 3,30
    J = 33-JJ
    IB = INDEX2(1,J)
    IB1 = 12*(IB-1)-11
    IB2 = 13 -IB
    WRITE(6,53) IB1,IB2,(TPN(I,J,K),I=IB,12)
71  CONTINUE
    DO 72 JJ = 2,31
    J = 33-JJ
    WRITE(6,54) (TPN(I,J,K),I=13,21)
72  CONTINUE
    DO 73 JJ = 3,30
    J = 33-JJ
    IE = INDEX2(2,J)
    IE1 = IE-21
    WRITE(6,55) IE1,(TPN(I,J,K),I=22,IE)
73  CONTINUE
    WRITE(6,1670)
    DO 74 JJ = 3,30
```

```
      J = 33-JJ
      IB = INDEX2(1,J)
      IB1 = 12*(IB-1)-11
      IB2 = 13 -IB
      WRITE(6,52) IB1,IB2,(QNG(I,J),I=IB,12)
   74 CONTINUE
      DO 75 JJ = 2,31
      J = 33-JJ
      WRITE(6,51) (QNG(I,J),I=13,21)
   75 CONTINUE
      DO 76 JJ = 3,30
      J = 33-JJ
      IE = INDEX2(2,J)
      IE1 = IE-21
      WRITE(6,50) IE1,(QNG(I,J),I=22,IE)
   76 CONTINUE

      IF(IMAP .EQ. 0 .OR. IMAP .EQ. 2 )GO TO 89

C  PLOTTING
      IF(ITRAN .EQ. 1) CALL TRPLOT(TSHUT,NTP,DEXX,DEYY,DEZZ)
      CALL OUTPLOT(T,TSHUT,ITRAN,ISCALE)

   89 IA = IA+1
      IF(IT .EQ. 2) GO TO 88
      IF(IA .GE. NT) GO TO 4999
      GO TO 1000
   88 IF(TIA .LT. TIME(2)) GO TO 1000
 4999 IF(IMAP .EQ. 1 .OR. IMAP .EQ. 3) CALL PLOT(0.,0.,999)

      REWIND1
      WRITE(1,14) TOI,QFL,ANG,T,DELT,CYCLE
      DO 65 J = 1,32
      IB = INDEX1(1,J)
      IE = INDEX1(2,J)
```

```
      DO 65 I = IB,IE
      WRITE(1,17)U(I,J,KD,V(I,J,KD,TP(I,J,KD,P(I,J,K)
   65 CONTINUE
    9 FORMAT("1",10A8,/////)
   11 FORMAT(4F10.4)
   13 FORMAT(10A8)
   14 FORMAT(5E15.8,I6)
   15 FORMAT("-","PROGRAM IS CONTINUOUS AT TIME = ",G12.5,"  CYCLE = ",
     1I6,"  DELT = ",G12.5,///)
   16 FORMAT(" ",T=,=F4.2)
   17 FORMAT(" ",4E15.8)
   18 FORMAT(10F8.3)
   19 FORMAT(" ",T40,"VOLUME FLOW RATE = ",G12.5,//,T25,"INLET VELOCITY"
     1,T55,"OUTLET VELOCITY",/,T14,"X-COMPONENT",T28,G12.5,/,
     2T58,G12.5,/,T14,"Y-COMPONENT",T28,G12.5,///)
   20 FORMAT(8I5)
   30 FORMAT(" ",2(8E15.8,/))
   31 FORMAT(16I5)
   32 FORMAT(F5.2,4(5E15.8,/))
   50 FORMAT(" ",=G12.4/)
   51 FORMAT(" ",9G12.4/)
   52 FORMAT(" ",T=,=G12.4/)
   53 FORMAT(" ",T=,=G12.5/)
   54 FORMAT(" ",9G12.5/)
   55 FORMAT(" ",=G12.5/)
   56 FORMAT(" ",2(T=,=G12.4/,1X)/)
   57 FORMAT(" ",2(9G12.4/,1X)/)
   58 FORMAT(" ",2(=G12.4/,1X)/)
   97 FORMAT(4I5,/)
 1001 FORMAT(//,5X,48H VELOCITY FIELD DOES NOT CONVERGE, PROGRAM STOPS ,
     1" CYCLE = " ,I4)
 1610 FORMAT(" ",T5,2I10,T37, G12.5,T57, G12.5,T75,F10.4,T85,F10.4,
     1T100,F10.4,T115,F10.4)
 1620 FORMAT("1",T12,"CYCLE",T23,"ITER",T42,"QLIN",T62,"QLOUT",
     1T80,"TAV",T90 ,"DELT",T102,"TI",T120,"TO")
```

```
1630  FORMAT("1",40X," A T  T I M E  =  ",G12.5,"SEC",//)
1640  FORMAT(" ",38X,"OR",G12.5,"  SECS,AFTER SHUT DOWN",/)
1650  FORMAT(" ",T60,"X-COMPONENT",/,T42,"VELOCITY MAP",/,T60,
     1"Y-COMPONENT",//)
1660  FORMAT("1",T60,"TEMPERATURE MAP"//)
1670  FORMAT("1",T55,"LOCAL HEAT SOURCE MAP (KW/CU MD" ,//)
5000  STOP
      END
```

```
      SUBROUTINE HEAT(QNG,POWR,POWM,IMOD)
      DIMENSION QNG(33,32),INDEX1(2,32),INDEX2(2,32),INDEX3(2,24),
     1PRTO1(120),PRTO2(4)
      DIMENSION ACTM(2),ARRY(2,20),IY(20),IZ(20),JY(20),JZ(20),DUM(20)
      COMMON /BLOCK3/ INDEX1,INDEX2,INDEX3,DELT,DEXX,DEYY,DEZZ
      COMMON /BLOCKB/ ACTM,ARRY,IY,IZ,JY,JZ,TSHUT,NZ,ITAP
      DATA PRTO1/.56,.56,.60,.63,.64,.53,.58,.66,.72,.77,.79,.8,
     1.55,.65,.72,.81,.87,.90,.92,.91,.57,.67,.78,.84,.92,.96,.98,.99,
     2.98,.55,.68,.78,.86,.91,.97,.98,.99,1.,1.,.50,.65,.77,.86,.91,
     3.93,.98,1.,1.,1.,1.,.6,.74,.86,.92,.94,.98,1.,1.,1.,1.,1.,.68,
     4.82,.91,.94,.98,1.,1.,1.,1.,1.,1.,.58,.73,.82,.96,.98,1.,1.,1.,1.,
     51.,1.,1.,.62,.78,.92,.99,1.,1.,1.,1.,1.,1.,1.,1.,.65,.82,.95,
     61.,1.,1.,1.,1.,1.,1.,1.,1.,.67,.84,.96,1.,1.,1.,1.,1.,1.,1.,1.,
     71./
      DATA PRTO2/.64,.80,.91,.98/
      POWR = 1000.*POWR
      POWM = 1000.*POWM
      VOL = DEXX*DEYY*DEZZ

      DO 1 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 1 I = IB,IE
      QNG(I,J) = POWR/246.
1     CONTINUE

      PMFCT = POWM/445.94
      N = 1
      DO 2 J = 5,16
      IB = INDEX3(1,J-4)
      JJ = 33-J
      DO 2 I = IB,16
      II = 34-I
      QNG(I,J) = PMFCT*PRTO1(N)
      N = N+ 1
```

```
      QNG(I,JJ) = QNG(I,J)
      QNG(II,J) = QNG(I,J)
      QNG(II,JJ) = QNG(I,J)
    2 CONTINUE

      N = 1
      DO 3 J = 5,8
      QNG(17,J) = PMFCT*PRTO2(N)
      N = N +1
      JJ = 33-J
      QNG(17,JJ) = QNG(17,J)
    3 CONTINUE
      DO 4 J = 9,24
      QNG(17,J) = PMFCT
    4 CONTINUE
      IF(IMOD .EQ.  0) RETURN

      IF(TSHUT .GT. ACTM(1) .AND. TSHUT .LE. ACTM(2)) GO TO 6
      READ(ITAP,32) A,(DUM(I),I=1,NZ)
      IF(EOF(ITAP)) 5,7
    7 ACTM(1) = ACTM(2)
      ACTM(2) = A
      DO 8 I=1,NZ
      ARRY(1,I) = ARRY(2,I)
      ARRY(2,I) = DUM(I)
    8 CONTINUE
      GO TO 6
    5 PRINT*,"POWER IS EXTRAPOLATED"
    6 FRACT = (TSHUT-ACTM(1))/(ACTM(2)-ACTM(1))
      QNE = POWM*VOL-3000.
      DO 12 L = 1,NZ
      DUM(L) = ARRY(1,L) + FRACT*(ARRY(2,L)-ARRY(1,L))
      JM = JY(L)
      JN = JZ(L)
      DO 12 J = JM,JN
```

```
      IM = IY(L)
      IN = IZ(L)
      IF(IM .LT. INDEX3(1,J-4)) IM = INDEX3(1,J-4)
      IF(IN .GT. INDEX3(2,J-4)) IN = INDEX3(2,J-4)
      DO 12 I = IM,IN
      F = QNG(I,J)/POWM
      QNG(I,J) = (QNE*F + DUM(L))/VOL
12    CONTINUE
32    FORMAT(F5.2,4(5E15.8,/))
      RETURN
      END
```

```
      SUBROUTINE HTEXCH(T2,TI,T3,QFA,TH2I)
      DIMENSION HTEMP(19),HVISC(19),HCOND(19),
     1TEMP1(19),VROR(19),CNDF(19),DECA(19),DDEN(19)
      COMMON /BLOCK8/ HTEMP,HVISC,HCOND,NDH
      COMMON /BLOCK9/ TEMP1,VROR,CNDF,DECA,DDEN,ND
      COMMON /BLOCKA/ TD20M,TH20,PSFLOW  ,Z2,Z3,Z5,Z6,Z7,
     1AX,DI,DO,AHI,PSF02,AHO,RC,RW
      QFL = QFA - 0.036
      T1 =(T2 + TI)/2.
      CALL LINT(DECA,TEMP1,CCP ,T1,ND)
      CALL LINT(VROR,TEMP1,VISC1,T1,ND)
      CALL LINT(CNDF,TEMP1,TCOND,T1,ND)
      CALL LINT(DDEN,TEMP1,DROR,T1,ND)
      CCP = CCP/DROR
      C1 = QFL*DROR*CCP
      PRN = CCP*VISC1/TCOND
      REZ4 = DROR/VISC1*0.51012E-6
      HI = TCOND/Z2*(  REZ4)**0.8*(PRN/Z3)**0.3*AHI
      RIO = DO/HI/DI
      CALL LINT(DDEN,TEMP1,DROR,T3,ND)
      CALL LINT(DECA,TEMP1,CP,T3,ND)
      CP = CP/DROR
      CALL LINT(HCOND,HTEMP,TCOND,T3,NDH)
      CALL LINT(HVISC,HTEMP,VISC1,T3,NDH)
      PRN = CP*VISC1/TCOND
      HO = TCOND/Z6*(PSFLOW*Z5/PSF02/VISC1)**0.6*(PRN/Z7)**0.3*AHO
      RO = 1./HO
      U = 1./(RIO + RO + RW + RC)
      C2 = PSFLOW*CP
      UTN = U*AX/C1
      CR = C1/C2
      PHI = EXP(UTN)
      P = 1./(CR*PHI**CR/(PHI**CR-1.)+PHI/( PHI-1.)-1./ALOG(PHI))
      IF(C1 .LT. C2) GO TO 751
      EEF = P*CR
```

```
      TH20 = TH2I + EEF*(T2-TH2I)
      TI = T2-C2/C1*(TH20-TH2I)
      T3 = (TH2I+TH20)/2.
      RETURN
751   EEF = P
      TI = T2-EEF*(T2-TH2I)
      TH20 = TH2I + C1/C2*(T2-TI)
      T3 = (TH2I + TH20)/2.
      RETURN
      END
```

```
      SUBROUTINE LINT(PT,T,PI,TI,N)
      DIMENSION PT(N),T(N)
      IF(TI .GE. T(1) ) GO TO 3
      I = 1
      GO TO 2
3     M = N-1
      DO 1 I = 1,M
      IF(TI .GE. T(I) .AND. TI .LT. T(I+1)) GO TO 2
1     CONTINUE
      I = M
      WRITE(6,10) TI,PI
2     PI = PT(I) +(TI-T(I))*(PT(I+1)-PT(I))/(T(I+1)-T(I))
10    FORMAT(" ", "WARNING** QUANTITY IS EXTRAPOLATED AT",F10.4,
     1"  AS",F10.4)
      RETURN
      END
```

```
      SUBROUTINE OUTPLOT (TPRT1,TPRT2,ITRAN,ISC)
      DIMENSION INDEX1(2,32),INDEX2(2,32),INDEX3(2,24),IS(2),UM(33,32),
     1ANGLE(33,32),TPN(33,32,1),QNG(33,32),FV(10),DUM(10),D(33,32)
      COMMON /BLOCK3/ INDEX1,INDEX2,INDEX3,DELT,DEXX,DEYY,DEZZ
      COMMON /BLOCK5/ TPN,QNG
      COMMON /BLOCK6/ UM,ANGLE,UREF
      DATA D/1056*41./
      DATA FV/45.,50.,55.,60.,65.,70.,75.,80.,85.,90./
      DATA IS/1,0/
      NCON = 10
      SIZE = 0.25
      XO = 3.
      YO = .5
      CALL PLOT(XO,YO,-3)
      CALL PLOT(21.*SIZE,SIZE,3)
      DO 1 L = 1,2
      DO 1 JJ = 2,31
      J = JJ
      IF(L .EQ. 2) J = 33-JJ
      XB = SIZE*(INDEX2(L,J)-IS(L))
      YB = SIZE*(J-IS(L))
      YE = SIZE*(J-L+1)
      CALL PLOT(XB,YB,2)
      CALL PLOT(XB,YE,2)
    1 CONTINUE
      XC = SIZE*29.
      CALL SYMBOL(XC,0.,0.14,8HSCALE = ,0.,8)
      IF(ISC .EQ. 0) CALL SYMBOL(XC+1.12,0.,0.14,6HLINEAR,0.,6)
      IF(ISC .EQ. 1) CALL SYMBOL(XC+1.12,0.,0.14,3HLOG,0.,3)
      IF(ISC .EQ. 2) CALL SYMBOL(XC+1.12,0.,0.14,3HNIL,0.,3)
      YC = 34.*SIZE
      CALL SYMBOL(0.,YC ,0.14,10HAT TIME = ,0.,10)
      IF(ITRAN .EQ. 1) TPRT1 = TPRT2
      CALL NUMBER(1.4,YC,0.14,TPRT1,0.,1)
      CALL SYMBOL(2.3,YC,0.14,3HSEC,0.,3)
```

```
      IF(ITRAN .EQ. 1) CALL SYMBOL(2.8,YC ,0.14, 16HAFTER BREAK DOWN,
     10.,16)
      DO 2 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 2 I = IB,IE
      UNOR = UM(I,J)/UREF
      IF(ISC .EQ. 0) GO TO 3
      UX = UM(I,J)*0.2/UREF
      UNOR = ALOG(0.2)/ALOG(UX)
    3 UMT = UNOR*SIZE*0.8
      IF(ISC .EQ. 2) UMT=SIZE*0.5
      XS = (I-0.5)*SIZE
      YS = (J-0.5)*SIZE
      AN = ANGLE(I,J)
      XR = XS + UMT*COS(AN)
      YR = YS + UMT*SIN(AN)
      TIP1X = XS + 0.75*UMT*COS(AN+0.053)
      TIP1Y = YS + 0.75*UMT*SIN(AN+0.053)
      TIP2X = XS + 0.75*UMT*COS(AN-0.053)
      TIP2Y = YS + 0.75*UMT*SIN(AN-0.053)
      CALL PLOT(XS,YS,3)
      CALL PLOT(XR,YR,2)
      CALL PLOT(TIP1X,TIP1Y,2)
      CALL PLOT(TIP2X,TIP2Y,3)
      CALL PLOT(XR,YR,2)
    2 CONTINUE

      CALL PLOT(XO,YO,-3)
C  CONTOURS PLOT
      DO 78 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 78 I = IB,IE
      D(I,J) = TPN(I,J,1)
```

```
78  CONTINUE
    ADJ = 0.5*SIZE
    CALL PLOT(21.5*SIZE,1.5*SIZE,3)
    DO 4 L = 1,2
    DO 4 JJ = 2,31
    J = JJ
    IF(L .EQ. 2) J = 33-JJ
    XB = SIZE*( INDEX2(L,J)-IS(L)) + ADJ
    YB = SIZE*(J-IS(L)) + ADJ
    YE = SIZE*(J-L+1) + ADJ
    CALL PLOT(XB,YB,2)
    CALL PLOT(XB,YE,2)
4   CONTINUE
    CALL PLTCON(D,33,32,FV,DUM,NCON)
    RETURN
    END
```

```
      SUBROUTINE TEMPFD
      DIMENSION U(33,32,1),V(33,32,1), T(33,32,1),P(33,32,1),
     1TPN(33,32,1),PO(34,33),INDEX1(2,32),INDEX2(2,32),
     2 INDEX3(2,24),QNG(33,32),VOR(33,32,1),CNF(33,32,1),DAC(33,32,1)
      COMMON /BLOCK1/ U,V,T,P
      COMMON /BLOCK3/ INDEX1,INDEX2,INDEX3,DELT,DEXX,DEYY,DEZZ
      COMMON /BLOCK4/ PO,VOR,CNF,DAC,RAD,TAV,TOI,BETA
      COMMON /BLOCK5/ TPN,QNG
      TAV = 0.
      K = 1
      DO 12 J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 12 I = IB,IE
      TUX =((U(I,J,K)+U(I-1,J,K)-ABS(U(I,J,K)+U(I-1,J,K)))*
     1(T(I+1,J,K)*DAC(I+1,J,K)-T(I,J,K)*DAC(I,J,K))+
     3   (U(I,J,K)+U(I-1,J,K)+ABS(U(I,J,K)+U(I-1,J,K)))*
     3(T(I,J,K)*DAC(I,J,K)-T(I-1,J,K)*DAC(I-1,J,K)))/4./DEXX
      TVY = ((V(I,J,K)+V(I,J-1,K)-ABS(V(I,J,K)+V(I,J-1,K)))*
     1(T(I,J+1,K)*DAC(I,J+1,K)-T(I,J,K)*DAC(I,J,K)) +
     2(V(I,J,K)+V(I,J-1,K) + ABS(V(I,J,K) + V(I,J-1,K)) *
     3(T(I,J,K)*DAC(I,J,K)-T(I,J-1,K)*DAC(I,J-1,K)))/4./DEYY
      TSX = ((CNF(I+1,J,K)+CNF(I,J,K))*(T(I+1,J,K)-T(I,J,K)) -
     1(CNF(I,J,K)+CNF(I-1,J,K))*(T(I,J,K)-T(I-1,J,K))
     2/2./DEXX**2
      TSY = ((CNF(I,J+1,K)+CNF(I,J,K))*(T(I,J+1,K)-T(I,J,K))-
     1(CNF(I,J,K)+CNF(I,J-1,K))*(T(I,J,K)-T(I,J-1,K))
     2/2./DEYY**2
      TPN(I,J,K) = T(I,J,K)+DELT/PO(I,J)/DAC(I,J,K)*(TSX+TSY+
     1QNG(I,J)-TUX-TVY)
      TAV = TAV + TPN(I,J,K)
   12 CONTINUE
      TAV = TAV/750.
      RETURN
      END
```

```
      SUBROUTINE TRPLOT(TSHUT,NTP,DEXX,DEYY,DEZZ)
      DIMENSION TEP(15),QEP(15),TEPI(15),QEPI(15),KVAL(15)
      DATA KVAL/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL,
     11HM,1HN,1HO/
      NQP = NTP
      CALL PLOT(1.5,1.5,-3)
      TMAX = QMAX = 0.
      REWIND 2
      READ(2,30) TEI
      READ(2,30) (TEPI(I),I=1,NTP)
      READ(2,30) (QEPI(I),I=1,NQP)

      DO 40 I = 1,NTP
      IF(TEPI(I) .GT. TMAX) TMAX = TEPI(I)
   40 CONTINUE
      DO 42 I = 1,NQP
      IF(QEPI(I) .GT. QMAX) QMAX = QEPI(I)
   42 CONTINUE
      VOL = DEXX*DEYY*DEZZ
      QMAX = QMAX*VOL
      SIZEX = 2.
      IF(TSHUT .GE. 4) SIZEX = 1./FLOAT(IFIX(TSHUT/8.+0.5))
      SIZEY = 1./FLOAT(IFIX(TMAX/6.+.5))
      SIZE = 1./FLOAT(IFIX(QMAX/6.+0.5))
      SIZEQ = SIZE*VOL

      CALL AXISN(0.,0.,8HTIME (S),8,1,0.5 ,8.,0.,0.,0.,0.50/SIZEX)
      CALL AXISN(0.,0.,8HTEMP (C),8,-1,0.5 ,6.,0.,90.,0.,0.50/SIZEY)
      CALL PLOT(TEI*SIZEX,TEPI(1)*SIZEY,3)
      WRITE(6,33)
      REWIND 2
      DO 1 I = 1,NTP
   50 READ(2,30) TIME
      IF(EOF(2)) 100,90
   90 READ(2,30) (TEP(J),J=1,NTP)
```

```
      READ(2,30) (QEP(J),J=1,NQP)
      WRITE(6,32) I,TIME,TEP(I),QEP(I)*VOL
      CALL PLOT(TIME*SIZEX,TEP(I)*SIZEY,2)
      GO TO 50
  100 CALL SYMBOL(TIME*SIZEX,TEP(I)*SIZEY,0.14,KVAL(I),0.,1)
      REWIND 2
      IF(I .EQ. NTP) GO TO 2
      CALL PLOT(TEI*SIZEX,TEPI(I+1)*SIZEY,3)
    1 CONTINUE

    2 CALL PLOT(1.5,1.5,-3)
      CALL AXISN(0.,0.,8HTIME (S),8,1,0.5 ,8.,0.,0.,0.,0.50/SIZEX)
      CALL AXISN(0.,0.,6HQ (KW),6,-1,0.5 ,6.,0.,90.,0.,0.50/SIZE)
      CALL PLOT(TEI*SIZEX,QEPI(1)*SIZEQ,3)
      DO 3 I = 1,NQP
   60 READ(2,30) TIME
      IF(EOF(2)) 110,105
  105 READ(2,30) (TEP(J),J=1,NTP)
      READ(2,30) (QEP(J),J=1,NQP)
      CALL PLOT(TIME*SIZEX,QEP(I)*SIZEQ,2)
      GO TO 60
  110 CALL SYMBOL(TIME*SIZEX,QEP(I)*SIZEQ,0.14,KVAL(I),0.,1)
      IF(I .EQ. NQP) GO TO 4
      CALL PLOT(TEI*SIZEX,QEPI(I+1)*SIZEQ,3)
      REWIND 2
    3 CONTINUE
   30 FORMAT(" ",2(8E15.8,/))
   32 FORMAT(" ",T10,I5,T20,3G20.5)
   33 FORMAT("1"///,T10,"LOCATION NO.",T31,"TIME",T51,"TEMP",T71,"HEAT"
     1/)
    4 RETURN
      END
```

```
      SUBROUTINE VELFD
      DIMENSION U(33,32,1),V(33,32,1),TP(33,32,1),P(33,32,1),
     1UN(33,32,1),VN(33,32,1),PO(34,33),
     2INDEX1(2,32),INDEX2(2,32),INDEX3(2,24),VOR(33,32,1),CNF(33,32,1),
     3DAC(33,32,1)
      COMMON /BLOCK1/ UN,VN,TP,P
      COMMON /BLOCK2/ U,V
      COMMON /BLOCK3/ INDEX1,INDEX2,INDEX3,DELT,DEXX,DEYY,DEZZ
      COMMON /BLOCK4/ PO,VOR,CNF,DAC,RAD,TAV,TOR,BETA
      K = 1
      DO 12  J = 2,31
      IB = INDEX2(1,J)
      IE = INDEX2(2,J)
      DO 12 I = IB,IE

      PX0 = (PO(I-1,J) + PO(I,J))/2.
      PX1 = (PO(I,J) + PO(I+1,J))/2.
      PX2 = (PO(I+1,J) + PO(I+2,J))/2.
      PY0 = (PO(I,J-1) + PO(I,J))/2.
      PY1 = (PO(I,J) + PO(I,J+1))/2.
      PY2 = (PO(I,J+1) + PO(I,J+2))/2.

      CX = 20.*DIM(PX1,0.95)*(VOR(I+1,J,K) + VOR(I,J,K))/2.
      CY = 20.*DIM(PY1,0.95)*(VOR(I,J+1,K) + VOR(I,J,K))/2.
      COEX = 16.67*CX*(1.-PX1)**2/PX1**3/RAD**2
      COEY = 16.67*CY*(1.-PY1)**2/PY1**3/RAD**2
      COX = 0.583  *(1.-PX1)/PX1**3/RAD
      COY = 0.583  *(1.-PY1)/PY1**3/RAD

      FUX = ((U(I+1,J,K)/PX2-U(I,J,K)/PX1)*(U(I,J,K)-ABS(U(I,J,K)))
     1 + (U(I,J,K)/PX1-U(I-1,J,K)/PX0)*(U(I,J,K)+ABS(U(I,J,K))))/2./DEXX
      VX = (V(I,J,K)+V(I,J-1,K)+V(I+1,J,K)+V(I+1,J-1,K))/4.
      UY = (U(I-1,J,K)+U(I-1,J+1,K)+U(I,J,K)+U(I,J+1,K))/4.
      FUY = ((2.*U(I,J+1,K)/(PO(I+1,J+1)+PO(I,J+1)) - U(I,J,K)/PX1)
     1  *(VX-ABS(VX))
```

```
     1 + (U(I,J,KD/PX1-2.*U(I,J-1,KD/(PO(I+1,J-1)+PO(I,J-1)))*(VX
     2 +ABS(VX)))/2./DEYY
      FVX = ((2.*V(I+1,J,KD/(PO(I+1,J+1)+PO(I+1,J))-V(I,J,KD/PY1)*
     1 (UY-ABS(UY)) + (V(I,J,KD/PY1-2.*V(I-1,J,KD/(PO(I-1,J)+
     2PO(I-1,J+1)))*(UY+ABS(UY)))/2./DEXX
      FVY = ((V(I,J+1,KD/PY2-V(I,J,KD/PY1)*(V(I,J,KD-ABS(V(I,J,KD))
     1 + (V(I,J,KD/PY1-V(I,J-1,KD/PY0)*(V(I,J,KD+ABS(V(I,J,KD)))/2./DEYY
      VISX = CX*((U(I+1,J,KD/PX2-2.*U(I,J,KD/PX1+U(I-1,J,KD/PX0)/DEXX**2
     1 + 2.*(U(I,J+1,KD/(PO(I+1,J+1)+PO(I,J+1)) -U(I,J,KD/PX1
     2 + U(I,J-1,KD/(PO(I+1,J-1)+PO(I,J-1)))/DEYY**2)
      VISY = CY*(2.*(V(I+1,J,KD/(PO(I+1,J+1)+ PO(I+1,J))-V(I,J,KD/PY1
     1 + V(I-1,J,KD/(PO(I-1,J+1)+PO(I-1,J)))/DEXX**2 + (V(I,J+1,KD/
     2PY2 - 2.*V(I,J,KD/PY1 + V(I,J-1,KD/PY0)/DEYY**2)
      UN(I,J,KD = U(I,J,KD + DELT*((P(I,J,KD-P(I+1,J,KD)/DEXX+PX1*(
     1    VISX - COEX*U(I,J,KD-SIGN(1.,U(I,J,KD)*COX*U(I,J,KD**2)-FUX
     2-FUY)
      VN(I,J,KD = V(I,J,KD + DELT*((P(I,J,KD-P(I,J+1,KD)/DEYY+PY1*(
     1    BETA*((TP(I,J+1,KD+TP(I,J,KD)/2. - TOR) + VISY -
     2 COEY*V(I,J,KD-SIGN(1.,V(I,J,KD)*COY*V(I,J,KD**2)-FVX-FVY)
  12  CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PLTCON(A,M,N,FV,DUM,NCON)
      DIMENSION A(M,N),FV(NCON),DUM(NCON),KK(11)
      COMMON/CONT/AL,BE,II,FI,SIZE
      DATA KK/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK/
      SIZE = 0.25
      AL = SIZE
      BE = SIZE
      DUM(1)=FV(1)
      DO 5 I=2,NCON
5     DUM(I)=FV(I)-FV(I-1)
      FI = 0.14
      J=0
      DO 6 I=1,NCON
      J=J+1
      II = KK(J)
      DO 22 IMO = 1,M
      DO 22 INO = 1,N
      A(IMO,INO) = A(IMO,INO)-DUM(I)
22    IF(A(IMO,INO) .EQ. 0.) A(IMO,INO) = 1.E-10
6     CALL HUNT(A,M,N)
      XB = 7.5
      CALL SYMBOL(3.,9.2,FI,14HTEMP. CONTOURS,0.,14)
      DO 7 I = 1,NCON
      YB = 9.3 - FLOAT(I)*(FI+0.07)
      CALL SYMBOL(XB,YB,FI,KK(I),0.,1)
      CALL SYMBOL(XB+FI,YB,FI,3H = ,0.,3)
      CALL NUMBER(XB+4.*FI,YB,FI,FV(I),0.,1)
      CALL SYMBOL(XB+9.*FI,YB,FI,1HC,0.,1)
7     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE HUNT(A,MM,N)
      COMMON /CONT/ AL,BE,KKK,FI,SIZE
      LOGICAL NOMORE, BOUND, FIRST
      COMMON/DETAIL/ KRIT,KI(33),KJ(33),KP(33)
      DIMENSION A(MM,N),KK(2)
      DATA M1,M2,M3,M4/177B,7777777777777777600B,7B,170B/
      DATA M5/4B/
      DATA KK/41,81/
      CALL TAIL(A,MM,N)
700   NPOS=3
      IS=ISCAN(A,MM,N,I,J)
      IF(IS.EQ.0) RETURN
      K=A(I,J).AND.M4
      A(I,J)=A(I,J).AND.M2
      FIRST=.TRUE.
      NOMORE=.TRUE.
      ASSIGN 210 TO IGO
      GO TO 400
210   NPOS=2
      FIRST=.FALSE.
      ASSIGN 220 TO IGO
220   IF(( L.AND.KK(3-NS).AND.M4).EQ.0) GO TO 300
      I=II
      J=JJ
225   IF((L.AND.M3).NE.1) GO TO 240
      A(I,J)=A(I,J).AND.M2
      K=L -1
      NOMORE=.TRUE.
      GO TO 400
240   IF(NS.EQ.1) GO TO 245
      IF((L.AND.8).EQ.0) GO TO 242
      K=8
      GO TO 250
242   K=32
      GO TO 250
```

```
245   IF((L.AND.16).EQ.0) GO TO 247
      K=16
      GO TO 250
247   K=64
250   LL=(L-K-1).AND..NOT.M5
      A(I,J)=(A(I,J).AND.M2).OR.(LL.AND.M1)
      NOMORE=.FALSE.
      GO TO 400
300   IF(NOMORE) GO TO 230
      IF((LL.AND.KK(3-NS).AND.M4).EQ.0) GO TO 230
      L=LL
      GO TO 225
230   IF(NS.EQ.2) GO TO 280
      IF(J.EQ.1) GO TO 260
      IF((A(I,J-1).AND.KO.EQ.0) GO TO 260
      J=J-1
270   L=A(I,J).AND.M1
      GO TO 250
260   IF(J.EQ.N) GO TO 600
      IF((A(I,J+1).AND.KO.EQ.0) GO TO 600
      J=J+1
      GO TO 270
280   IF(I.EQ.1) GO TO 290
      IF((A(I-1,J).AND.KO.EQ.0) GO TO 290
      I=I-1
      GO TO 270
290   IF(I.EQ.MM) GO TO 600
      IF((A(I+1,J).AND.KO.EQ.0) GO TO 600
      I=I+1
      GO TO 270
600   IF(BOUND) GO TO 650
      IF(ABS(X-XS).GT.AL.OR.ABS(Y-YS).GT.BE) GO TO 650
      CALL PLOT(XS,YS,NPOS)
      GO TO 700
 650  CALL SYMBOL(X*SIZE,Y*SIZE,FI,KKK,0.,1)
```

```
      GO TO 700
400   IF(K.NE.8) GO TO 420
      II=I-1
      X=FLOAT(I)+A(I,J)/(A(II,J)-A(I,J))
      GO TO 450
420   IF(K.NE.16) GO TO 440
      JJ=J-1
      Y=FLOAT(J)+A(I,J)/(A(I,JJ)-A(I,J))
      GO TO 470
440   IF(K.NE.32) GO TO 460
      II=I+1
      X=FLOAT(I)+A(I,J)/(A(I,J)-A(II,J))
450   JJ=J
      NS=1
      Y=FLOAT(J)
      GO TO 500
460   JJ=J+1
      Y=FLOAT(J)+A(I,J)/(A(I,J)-A(I,JJ))
470   II=I
      NS=2
      X=FLOAT(I)
 500  X = X*SIZE
      Y = Y*SIZE
      IF(.NOT.FIRST) GO TO 520
      BOUND=(IS.AND.NS).NE.0
      IF(.NOT.BOUND.AND.(IS.NE.4)) GO TO 800
      XS=X
      YS=Y
      CALL SYMBOL(X,Y,FI,KKK,0.,1)
520   CALL PLOT(X,Y,NPOS)
      L=A(II,JJ).AND.M1
      L=L+K-KK(NS)
      L=L.AND..NOT.M5
      A(II,JJ)=(A(II,JJ).AND.M2).OR.(L.AND.M1)
      IF (KRIT.EQ. 0) GO TO IGO,(210,220,900)
```

```
      IF(NS.EQ.1) GO TO 530
      J1=(J+JJ+1)/2
      DO 540 M=1,KRIT
      IF(J1.NE.KJ(M)) GO TO 540
      IF(I.EQ.KI(M)) GO TO 543
      IF(I+1.EQ.KI(M)) GO TO 546
  540 CONTINUE
      GO TO IGO,(210,220,900)
  543 K=8
      J=KJ(M)-KP(M)
      GO TO 560
  546 K=32
      J=KJ(M)+KP(M)-1
  560 K2=16
      IF(J.EQ.KJ(M)) K2=64
      GO TO 590
  530 I1=(I+II+1)/2
      DO 550 M=1,KRIT
      IF(I1.NE.KI(M)) GO TO 550
      IF(J.EQ.KJ(M)) GO TO 553
      IF(J+1.EQ.KJ(M)) GO TO 556
  550 CONTINUE
      GO TO IGO,(210,220,900)
  553 K=16
      I=KI(M)-KP(M)
      GO TO 570
  556 K=64
      I=KI(M)+KP(M)-1
  570 K2=8
      IF(I.EQ.KI(M)) K2=32
  590 ASSIGN 900 TO IGO
      LL=(A(I,J).AND.M1)-K-1
      LL=LL.AND..NOT.M5
      A(I,J)=(A(I,J).AND.M2).OR.LL
      KI(M)=KI(KRIT)
```

```
      KJ(MD)=KJ(KRIT)
      KP(MD)=KP(KRIT)
      KRIT=KRIT-1
      NOMORE=.FALSE.
      GO TO 400
900   ASSIGN 220 TO IGO
      IF((L.AND.K2).EQ.0) GO TO 910
      I=II
      J=JJ
      K=K2
      GO TO 250
910   IF((LL.AND.K2).EQ.0) GO TO 920
      L=LL
      K=K2
      GO TO 250
920   IF(NS.EQ.2) GO TO 930
      IF(K2.EQ.64) GO TO 260
      IF(J.EQ.1) GO TO 600
      J=J-1
      IF((A(I,J).AND.K).EQ.0) GO TO 600
      GO TO 270
930   IF(K2.EQ.32) GO TO 290
      IF(I.EQ.1) GO TO 600
      I=I-1
      IF((A(I,J).AND.K).EQ.0) GO TO 600
      GO TO 270
800   A(I,J)=A(I,J).OR.M5.OR.K
      GO TO 700
      END
```

```
      FUNCTION ISCAN(A,M,N,II,JJ)
      DIMENSION A(M,N)
      DATA M1/7B/
      J=1
      DO 50 I=1,M
      IF((A(I,J).AND.M1).EQ.1) GO TO 100
   50 CONTINUE
      I=M
      DO 60 J=2,N
      IF((A(I,J).AND.M1).EQ.1) GO TO 100
   60 CONTINUE
      J=N
      MM=M+1
      DO 70 K=2,M
      I=MM-K
      IF((A(I,J).AND.M1).EQ.1) GO TO 100
   70 CONTINUE
      I=1
      NN=N+2
      DO 80 K=3,N
      J=NN-K
      IF((A(I,J).AND.M1).EQ.1) GO TO 100
   80 CONTINUE
      MM=M-1
      NN=N-1
      DO 90 J=2,NN
      DO 90 I=2,MM
      IF((A(I,J).AND.M1).EQ.1) GO TO 100
   90 CONTINUE
      ISCAN=0
      RETURN
  100 ISCAN=4
      IF(I.EQ.1.OR.I.EQ.M) ISCAN=6
      IF(J.EQ.1.OR.J.EQ.N) ISCAN=ISCAN+1
      II=I
```

```
      JJ=J
      RETURN
      END
```

```
      SUBROUTINE TAIL(A,M,N)
      DIMENSION A(M,N)
     COMMON/DETAIL/ KRIT,KI(33),KJ(33),KP(33)
      DATA M1/177B/
      M2=.NOT.M1
     KRIT=0
     DO 100 I=1,M
     DO 100 J=1,N
      K=0
     T=SIGN(1.0,A(I,J))
      IF(I.EQ.1) GO TO 20
     IF (SIGN(1.0,A(I-1,J)) .NE. T) K=K+9
      IF(I.EQ.M) GO TO 30
20   IF (SIGN(1.0,A(I+1,J)) .NE. T) K=K+33
30    IF(J.EQ.1) GO TO 40
     IF (SIGN(1.0,A(I,J-1)) .NE. T) K=K+17
      IF(J.EQ.N) GO TO 50
40   IF (SIGN(1.0,A(I,J+1)) .NE. T) K=K+65
50    A(I,J)=(A(I,J).AND.M2).OR.K
     IF ((A(I,J) .AND. 24) .NE. 24) GOTO 100
     IF(I.EQ.1) GO TO 100
     IF(J.EQ.1) GO TO 100
     IF ((A(I-1,J-1) .AND. 96) .NE. 96) GOTO 100
     IF(KRIT .EQ. 33) GO TO 100
     KRIT=KRIT+1
     KI(KRIT)=I
     KJ(KRIT)=J
     T=ABS(A(I,J)+A(I-1,J-1))-ABS(A(I-1,J)+A(I,J-1))
     K=0
     IF(T.GT.0.0) K=1
     KP(KRIT)=K
100  CONTINUE
      RETURN
      END
      END
```

APPENDIX B

Heat Exchangers Model

B.1 Introduction

The transient behaviour of the moderator heat exchangers is modelled using the effectiveness - NTU method. The temperature relationships and heat exchangers effectiveness applicable for shell and tube heat exchangers having split flow shells and multi pass tubes are given by Chapman (4) and Jaw (5). The following model has been described by Keil (6).

B.2 Model

The heat exchangers relate the temperature of the inlet and outlet of the moderator. The outlet fluid temperature of the heat exchangers depend on their thermal effectiveness, temperature of the service and inlet fluid.

If the tube side thermal capacitance is less than that in the shell side, then

$$T_{D_2O,\,out} = T_{D_2O,in} + P\cdot(T_{D_2O,in} - T_{H_2O,in}) \qquad B\text{-}1$$

where P is the thermal effectives, and if the shell side thermal capacitance is less than that of the tube side, then

$$T_{H_2O,out} = T_{H_2O,in} + P\cdot R\cdot(T_{D_2O,in} - T_{H_2O,in}) \qquad B\text{-}2$$

$$T_{D_2O,out} = T_{D_2O,in} + 1/R\cdot(T_{H_2O,out} - T_{H_2O,in}) \qquad B\,3$$

where the dimensionless group R is defined as

$$\frac{\dot{M}_{D_2O}\cdot C_{D_2O}}{\dot{M}_{H_2O}\cdot C_{H_2O}} \qquad B\,4$$

Value of T_{D_2O}, out is calculated either from eq B-1 or eq B-2 and B-3

The heat exchanger effectiveness, P, obtained from Jaw (5) is written as

$$P = 1./(R\cdot\phi^R/(\phi^R-1) + \phi/(\phi-1) - 1./\ln\phi) \qquad B\,5$$

where $\phi = \exp\ (U \cdot A / \dot{M}_{D_2O} \cdot C_{D_2O})$

A total outside tube heat transfer surface area (m^2)

and

$$U = \frac{1}{\frac{1}{h_i} + \frac{1}{h_o} + R_w + R_c} \qquad \text{B·6}$$

where hi is the tube side heat transfer coefficient (kw/m^2C)

ho is the shell side heat transfer coefficient (kw/m^2C)

Rw is the thermal resistance f the tube material (m^2/kw)

Rc is the thermal resistance due to fouling (m^2C/kw)

The value of hi is obtained using the correlation given by Chapman (4) for turbulent flow through tubes.

$$h_i = \frac{k_i}{Z_2} \left[\frac{N_{Re}}{Z_4}\right]^{.8} \left[\frac{N_{PR}}{Z_3}\right]^{.03} \cdot AHI \qquad \text{B·7}$$

where ki is the thermal conductivity of D_2O at Ti

Ti is the mean D_2O temperature

Z_2 is the thermal conductivity of D_2O at design temperature

N_{RE} is the Reynold number evaluated at Ti

Z_4 is the Reynold number evaluated at design temperature

N_{PR} is the Prandtl number of D_2O at Ti

Z_3 is the Prandtl number of D_2O at the design temperature

AHI is the heat transfer coefficient for design conditions

A correction to the outside surface area is made to hi

$$h_i = h_i \cdot d_i / d_o$$

where di is the inside diameter of the tubes

do is the outside diameter of the tubes

The value of ho is obtained for flow across bundles of tube

$$h_o = \frac{k_o}{Z_6}\left[\frac{\dot{M}_{H_2O}\dot{\mu}_{H_2O}}{\dot{M}'_{H_2O}\mu_{H_2O}}\right]^{0.6}\left[\frac{N_{PR}}{Z_7}\right]^{0.33} \cdot AHO \qquad B\cdot 8$$

where ko is the thermal conductivity of H_2O at To

To is the mean H_2O temperature

Z_6 is the thermal conductivity of H_2O at design temperature

$\dot{M}_{H_2O}$ is the mass flow rate at To

$\dot{M}'_{H_2O}$ is the mass flow rate at design temperature

μ is the viscosity of water at To

μ' is the viscosity of water at design temperature

N_{PR} is the Prandtl number of H_2O at To

Z_7 is the Prandtl number of H_2O at design temperature

AHO is the heat transfer coefficient for design conditions

Equation B-8 is recommended for $N_{RE} > 4000$

Appendix C

Temperature Contours Plot

C.1 Introduction

The contours plot package includes subprograms, PLTCON, HUNT, ISCAN, TAIL which is compatible with MODCONV to produce isothermals inside the moderator.

C.2 Basic Philosophy

We are always looking for the contour with value zero. It is traced through points on lines joining T(I,J) to one of T(I-1,J), T(I+1,3), T(I,J-1), T(I,J+1), and these points are joined by straight line segments. A contour crosses a line only if the two end values have opposite signs.

The seven low order bits of each array value are set as follows: i) last 3 bits - indicator count for the number of nearest neighbour points sign, ii) next 4 bits -

indicators refer to the 4 nearest points are set to 1 if that point has opposit sign. Compass routine TAIL sees indicators and indicator count. We start a contour from a point with just one indicator set or those cross the boundary. Compass routine ISCAN finds a starting point. A contour is continued as follows. Each time we cross a line joining 2 points, we delete the appropriate indicators in these two array values and reduce their indicator counts by one. These two points are referred to as new and old. We then try to find i) A line through new perpendicular to the last line crossed, ii) a line through old perpendicular to the last line crossed, iii) a line parallel to the last line crossed. We use the first of these that is available. If none of these 3 methods can continue the contour, we check to see whether it is an internal contour and should be joined to its starting point or not. We then return to ISCAN to see if we have another contour. To find the precise point on those line we linearly interpolate for zero from the 2 array values at each end.

C-3 Input

Values of isothermals are specified in the DATA statement for FV.

Note: Subprograms, PLTCON, HUNT, ISCAN, TAIL are modified subroutines from contour plotting package by Dr. D.J. Kenworthy in McMaster University.

Appendix D

JCL and CALCOMP Plotting Routines

A.

The basic system call concept is shown in the block diagram D-1. The fortran deck and complier can be by-passed, if the card deck is stored in a complied form. Basically, the final result is obtained through two sources, i) line printer, ii) x-y plotter.

The following is the typical JCL used.

```
MACHINE C.
JOB, CM1Ø5ØØØ P3.
ACCOUNT, YTC, Ø38, A259, C811.
NOTE Y.T. CHAN.
DISPOSE, OUTPUT, *LP, OH.
GET, TAPE1 : filename1.
FTN, A, OPT : 2.
LDSET, LIB : CALCOMP.
LGO.
REPLACE, TPE1 : filename2.
DISPOSE, TAPE99, *MB : filename3.
EXIT.
```

FTN statement may be replaced by GET, LGO : YTCBIN if the code is stored in YTCBIN in compiled form.

The plotting information which is calculated by MODCONV is disposed on tape99 through CALCOMP plotting library routine into the TCL system. Thus, it is rather important to lay out a proper procedure to obtain the plots.

Dail 9-449-1241 to multiple access computer system, then type

```
ENTER, PROJ, USER, CU - ytc, Ø38, a259  (login TCL system)
ENTER, SEC
     C811
TERM : 53, TTY
TYPE SYSTEM NAME - tcl
     use, scan99                        (access ALCOMP routine)
*SCAN99 *VERSION2 - CALCOMP PLOT FILE PREVIEW
ENTER COMMAND ? plot, yttest            (plot from tape yttest)
ENTER COMMAND ? xyplot                  (access x-y plotter)
ENTER COMMAND ? post, off               (no postscaning)
ENTER COMMAND ? auto, off               (no autowindow size set)
ENTER COMMAND ? step                    (stepwise mode)
ENTER COMMAND ? window, 15,10           (set window size to
                                         15"x10")
ENTER COMMAND ? window, 0,0             (set origin to the center
                                        of the window)
```

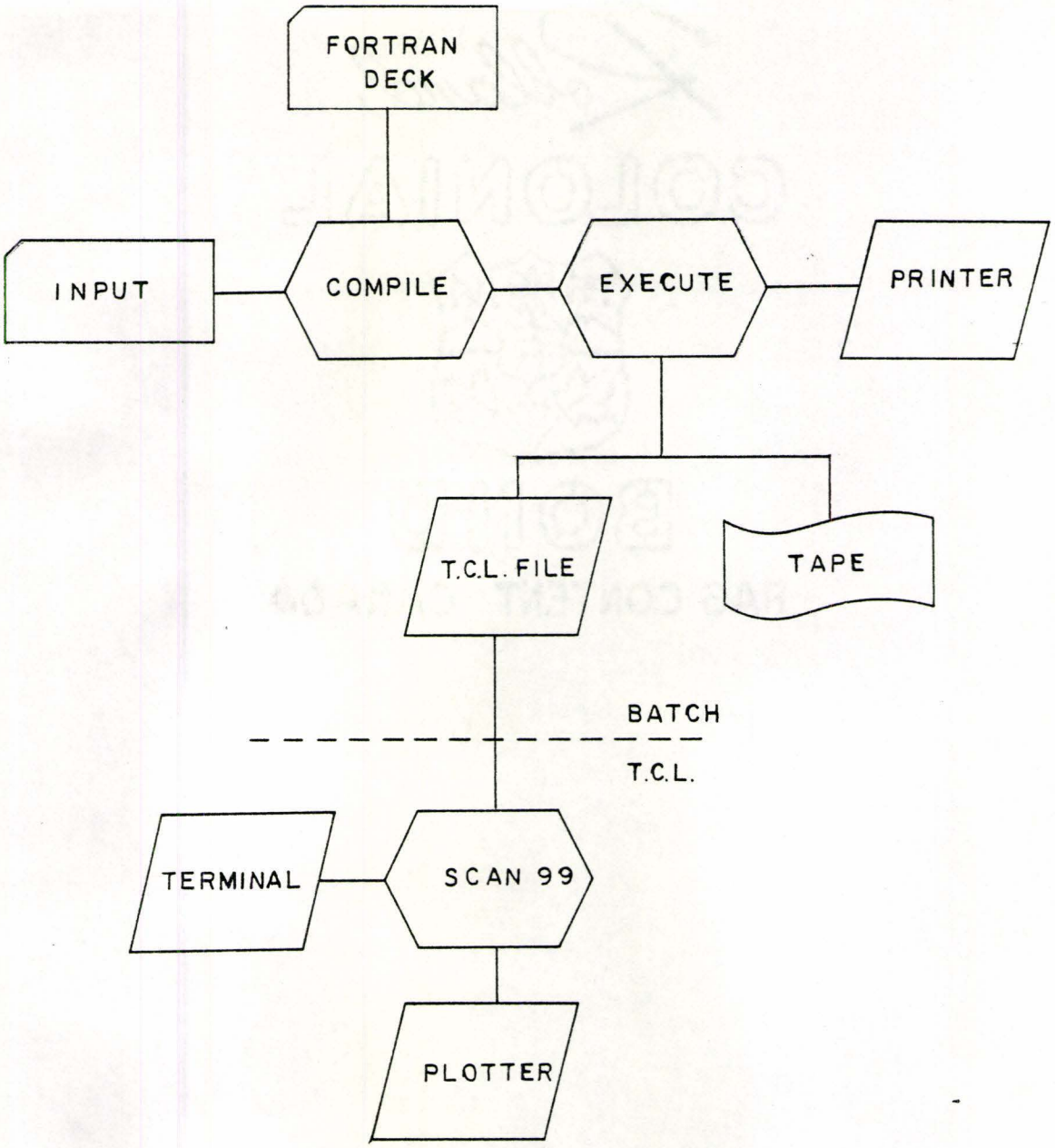
ENTER COMMAND ? I (print parameter settings)

ENTER COMMAND ? go (print date, jobname, account number of the disposing job. This printing can be avoided by either unload the plotter or used Advance 1, or search 3)

ENTER COMMAND ? go (each 'go' will allows one plotting)

An 'end' is required to sign off the SCAN99 routine, while a 'logout', 'bye' or 'hello' can be used to sign off the TCL.

NOTE: Sometimes, the origin may need to redefined between plots. By default, the scale is 1. The scale can be changed with comand SCALE, N.


FORTRAN DECK
INPUT
COMPILE
EXECUTE
PRINTER
T.C.L. FILE
TAPE
BATCH
T.C.L.
TERMINAL
SCAN 99
PLOTTER