NBSIR 83-2804

Thermal Flanking Loss Calculations for the National Bureau of Standards Calibrated Hot Box

R. J. Onega

Department of Mechanical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

and

P. J. Burns

Department of Mechanical Engineering
Colorado State University
Fort Collins, CO 80523

February 1985

Prepared for
Department of Energy
Oak Ridge National Laboratory
Energy Division
Oak Ridge, TN 37830

NBSIR 83-2804

THERMAL FLANKING LOSS CALCULATIONS FOR THE NATIONAL BUREAU OF STANDARDS CALIBRATED HOT BOX

R. J. Onega

Department of Mechanical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

and

P. J. Burns

Department of Mechanical Engineering
Colorado State University
Fort Collins, CO 80523

February 1985

Prepared for
Department of Energy
Oak Ridge National Laboratory
Energy Division
Oak Ridge, TN 37830

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

FORWORD

This report is one of a series documenting NBS research and analysis efforts in support of the Department of Energy/Oak Ridge National Laboratory/National Bureau of Standards' Building Thermal Envelope Systems and Insulation Materials program. The work reported in this document was performed under the Calibrated Hot Box project (Task 4) supported by DOE/NBS Interagency Agreement No. DE-AI05-780R06113.

EXECUTIVE SUMMARY

Two frame coefficients are defined FCMCA and FCHXS which describe the fraction of the heat lost through the frame. FCMCA is related to the actual heat flux that goes through the wall specimen and FCHXS is related to the heat flux that goes through the specimen if it was one-dimensional.

The code FLANK (see Appendix) was written to obtain these coefficients, the temperature distribution and the heat fluxes across various surfaces in both a steady state or transient mode. The code is a two dimensional, finite difference code. The number of nodes that can be used is variable and is limited only by computer memory and the amount of time and money available for computer runs.

If one considers the steady-state conditions for the metering chamber, the heat input to the metering chamber Q_{IN} is related to the heat out by $Q_{IN} = Q_{specimen} + Q_{frame} + Q_{MC\ walls}$ where Q_{frame} is the heat that goes out through the frame and $Q_{MC\ walls}$ is the heat lost through the metering chamber walls. It is easily seen that

$$Q_{frame} = Q_{IN} \left(1 - \frac{Q_{specimen}}{Q_{IN}}\right) - Q_{MC\ walls}$$

$$= Q_{IN}\ \mathrm{FCHXS} - Q_{MC\ walls}$$

if the heat through the specimen is assumed to be $k_s\ A_s\ (T_{EC1}-T_{MC1})\Delta Y_s$ where k_s and A_s are the thermal conductivity and area of the specimen, ΔY_s is the thickness of the specimen and T_{EC1} and T_{MC1} are the surface temperatures at the center of the climatic and metering chamber sides of the wall. FLANK provides FCHXS (and FCMCA) for a given set of conditions.

Typical applications for a polystyrene wall indicate that FCHXS will be between 5 and 15 percent in steady-state applications. If the temperature of the climatic chamber is equal to the temperature of the metering chamber, it can be seen that the frame coefficients will be very large (≈1). In the transient case, the polyurethane will take about 40 or 50 hours to reach steady state. The frame coefficients likewise will achieve their steady state values in about 50 hrs or so. Also, the frame coefficients can be very large (≈50 percent) in the transient situation due to the fact that the polyurethane frame itself is being heated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 INTRODUCTION

A computer code called FLANK (see Appendix) was written to calculate the flanking loss for the NBS Calibrated Hot-Box (CHB). This code is a two-dimensional finite difference dynamic thermal simulation of the test frame and specimen when subjected to either steady-state or time-varying boundary conditions. The code is specifically designed to do the calculations for the NBS-CHB but the code is general enough to be used with any CHB.

This report presents some background information of the NBS-CHB since the code was written so that those quantities which are measurable or controllable are either inputs or outputs of the code. Following this is a description of the mathematical model, a description of the code and results. A users manual for FLANK is also available.

1.2 BACKGROUND

The purpose of the NBS Calibrated Hot-Box (CHB) is primarily to measure the heat flow through large components of a building envelope, such as a wall. A test specimen is constructed in a frame which is then placed between two chambers. One chamber is called the metering chamber (MC) and is designed to simulate inside conditions and the remaining chamber is called the environmental chamber (EC) which simulates outside conditions. Wider temperature variations are possible in the EC than the MC.

The following purposes can be enumerated for the NBS-CHB:

1. Measure the heat, air and moisture transport through large homogeneous or composite wall system under steady-state and dynamic conditions.

2. Provide the methodology in CHB measurements that can be used for accrediting various testing laboratories.

3. Generate thermal transport properties for full scale wall systems which may include windows, doors, etc.

4. Provide a mechanism for traceability to NBS in the measurement of thermal properties of wall specimens through a concept such as the Standard Reference Material (SRM) program.

5. Verify models of heat, moisture, air transport and flanking loss through walls by comparing model results to experimental data.

A flanking loss correction must be made to provide a comparison of theory and experiment. The flanking loss is a function of the wall thickness, material, placement of the wall in the frame, the test frame construction and the temperature in the MC, EC and the environment. Since air temperatures, heat transfer coefficients, etc., can be experimentally imposed, the model must be able to accommodate these variables as inputs.

1.3 THE NBS-CHB SPECIFICATIONS

The CHB is to test walls with a thermal resistance ranging from an R of 0.352 $m^2 \cdot K/W$ to 8.81 $m^2 \cdot K/W$ (R-2 to R-50 $hr \cdot ft^2$ °F/Btu) under a variety of temperature, moisture and air infiltration conditions. The MC Walls are made of polyurethane board about 0.38 m (15 in) thick surrounded on all outside surfaces by a water jacket designed to minimize the temperature difference between the outside and inside of the walls. The interior and exterior surfaces of the polyurethane MC walls are covered with about 0.003175 m (~1/8 in) fiberglass reinforced plastic (FRP) of type 45 fiberglass polyester 45 percent glass matt base with a fire-retardant filler. Specifics of the wall construction construction are presented in table 1.1.

The EC walls consist of 0.454 m (17-3/4 in) polyurethane covered with FRP. The FRP is finished with two coats of epoxy-base paint. The R-value for the EC walls are about 19.02 $m^2 \cdot K/W$ (R - 108 h $ft^2 \cdot$°F/Btu).

The wall specimen is mounted in a polyurethane frame. Details of the frame and specimen are presented in table 1.2. The wall specimen itself is 3.048 m high (10 ft), 4.572 m wide (15 ft) and can vary in thickness up to a maximum of 0.6096 m (2 ft). Neoprene seals are used to sandwich the test frame between the MC and the EC.

Table 1.3 provides a list of operating conditions available in both the MC and the EC.

1.4 SUMMARY OF TESTS TO BE RUN

Various test specimens will be put into the CHB and subjected to various climatic conditions. These walls may be homogeneous or composites and may contain windows and doors. Table 1.4 is taken from reference [1]. Three different types of parameters can be varied: a) thermal quantities - air temperatures in both chambers; b) air infiltration parameters - static pressure difference across the wall; c) moisture parameters - relative humidity in both chambers. These parameters determine the heat flow through a wall with given thermal characteristics [2]. Both static and dynamic tests are to be run.

The control systems are to maintain constant dry-bulb temperatures to within $\pm$ 0.11 K of their setpoints in both chambers. A static pressure differential across the test specimen is to be maintained to within $\pm$ 2.5 Pa of its set point. The air flow rates can be maintained constant and will generally be directed in the direction that natural convection would take place.

1.5 CHB SENSORS

The results of computer models must be compared with experimental data to determine their accuracy and range of validity. It is, therefore, important to know what sensors are used, their accuracy and precisely what quantity is being measured. Table 1.5 provides a list of these sensors. The sensors in the specimen itself will depend upon the tests and the specimen.

Table 1.1 Specifications of the Walls of the CHB

QUANTITY	METERING CHAMBER	ENVIRONMENTAL CHAMBER
Outside height	4.3 m	4.3 m
Outside length	6.7 m	6.7 m
Outside width	1.50 m	1.57 m
Inside width	1.117 m	1.117 m
Inside height	3.048 m	3.048 m
Inside length	4.572 m	4.572 m
Wall thickness	0.38 m	0.4572 m
Wall construction	polyurethane	polyurethane
Inside volume	15.6 m^3	15.6 m^3
Thermal resistance	-	19.0 $m^2 \cdot K/w$

Table 1.2 Test Specimen and Specimen Frame Specification

QUANTITY	VALUE
Construction	Rigid polyurethane foam covered with a layer 0.003175 m of fiberglass
R-Value	Reinforced plastic (FRP) 0.352 to 8.81 $m^2 \cdot K/w$
Inside height	3.048 m
Inside width	4.572 m
Frame width	2 frames (0.3048 m and 0.6096 m)
Frame thickness	0.4572 m
Specimen area	13.935 m^2
Specimen weight/area	700 kg/m^2
Air leakage rate (max.)	255 m^3/hr
Moisture transfer rate (max.)	3.89 kg/s
Static pressure diff. (max.)	125 Pa

Table 1.3 Operating Conditions That May be Imposed on a Specimen in the NBS-CHB

PARAMETER	METERING CHAMBER	ENVIRONMENTAL CHAMBER
Dry bulb temperature range	283 to 338 K	233 to 338 K
Dew point temperature range	277.4 at 283 K D.B. to 293 at 297 K D.B. and above	230 at 233 K D.B to 293 K D.B. and above
Heat input rate (max.)	7.3 kW	12.3 kW
Heat removal rate (max.)	6.4 kW	12.3 kW
Velocity of air curtain	0.25 to 0.75 m/s	0.41 to 4.1 m/s
Maximum diurnal temperature amplitude	17°C	56°C
Rate of temperature rise at 27°C (min.)	3.06×10^{-3} °C/s	3.06×10^{-3} °C/s
Rate of temperature decrease at -7°C	2.72×10^{-3} °C/s	2.22×10^{-3} °C/s

Table 1.4 Climatic Conditions to be Simulated

SEASON	Thermal Direction[a]	Thermal S.S or Dyn[b]	Air Direction[a]	Air S.S or Dyn[b]
			Transmission Phenomena	
Winter	Out	S.S	None[c]	
Winter	Out	Dyn	None	
Summer	In	S.S	None	
Summer	In	Dyn	None	
Winter	Out	S.S	Recirc[e]	S.S
Summer	In	S.S	Recirc	S.S
Winter	Out	S.S	Recirc	S.S
Winter	Out	Dyn	Recirc	Dyn
Summer	In	Dyn	Recirc	Dyn
Winter	Out	S.S	In	S.S
Winter	Out	S.S	Out	S.S
Summer	In	S.S	In	S.S
Summer	In	S.S	Out	S.S
Winter	Out	Dyn	In	S.S
Summer	In	Dyn	In	S.S
Winter	Out	Dyn	Out	S.S
Summer	In	Dyn	Out	S.S
Winter	Out	S.S	Out	S.S
Winter	Out	Dyn	Out	S.S

a. "In" and "Out" refer to the direction of heat, air, or moisture transfer as viewed from the Metering Chamber.

b. "S.S" stands for steady-state. "Dyn" means dynamic or variable.

c. "None" means no static pressure difference across the specimen.

d. "None" means that no vapor pressure difference will be maintained across the specimen.

e. "Recirc" means two-way flow of air can take place through some wall specimens due to stack effect.

Table 1.5 Sensors in the NBS-CHB

QUANTITY	METERING CHAMBER (ACCURACY)	ENVIRONMENTAL CHAMBER (ACCURACY)
1. Temperature a) specimen surface temperature	40, 24 gage Cu-Co thermocouples - type T ($\pm$ 0.5 K)	40, 24 gage type T thermocouple ($\pm$ 0.5 K)
b) air temperature i) dry bulb	RTD	RTD
ii) dew point temperature		
2. Heat flux	30 heat-flow meters flat, 2-1/4", installed between polyurethane and FRP in interior of chamber	same as M.C.
3. Pressure		
4. Power input a) heat to metering chamber	($\pm$ 0.5 percent of reading)	
5. Air velocity		

1.6 SUMMARY OF THERMOPHYSICAL DATA

Steady-state and dynamic modeling require knowledge of thermophysical parameters. Table 1.6 is a list of the relevent parameters for some of the materials from which the CHB is constructed. In addition to thermophysical parameters, air infiltration and moisture parameters are also required.

1.7 FLANKING LOSS DEFINITION (Frame Coefficient)

The flanking loss is loosely defined as the amount of heat that leaves the MC which does not go through the wall and ends up in the EC. Lavine et al. [4] define the flanking loss as "heat flow also occurs between the metering chamber and the climatic chamber through the specimen frame. This heat flow is called the flanking loss."

According to the definition of Lavine, et al., the flanking loss is only that heat which flows from the MC, through the frame and then into the EC.

Table 1.6 Nominal Thermophysical Parameters for CHB Construction Materials and Some Specimen Materials

Material	-k (W/m• K)	$C_p(\frac{J}{kg.K})$	$\rho(\frac{kg}{m^3})$	$\alpha(\frac{m^2}{s})(\frac{k}{\rho c})$
1. Polyurethane foam	0.023	921.1	32.0-46.5	7.79×10^{-7}
2. Fiberglass reinforced plastic (FRP)	0.43 (anisotropic)	935	1400-2000	3.28×10^{-7}
3. Neoprene	0.180	2000	~1100	$\sim 8 \times 10^{-8}$
4. Polystyrene				
a) expanded bead	0.0288	1214	16.02	1.48×10^{-6}
b) expanded bead	0.0345	1214	24.03	1.18×10^{-6}
c) expanded bead	0.0332	1214	32.04	8.54×10^{-7}
5. Fiberglass (dry)				
a) light	0.0389	778	12.0	4.17×10^{-6}
b) medium	0.0332	775.0	48.0	8.92×10^{-7}

This definition result in several questions which are listed below:

i) How is the heat that flows through the edge of the test wall accounted for in this definition?

ii) How is the heat that flows from the MC, through the frame and out into the surroundings accounted for in this definition?

iii) How does one uniquely calculate flanking loss as defined above?

Because of these difficulties, the following definitions of a quantity termed "frame coefficient" are preferred. Both of these definitions depend upon the deviation from one-dimensional heat flow for their basis.

The total heat flow out of the MC (for steady-state) is observed from the difference between the amount produced from the electrical heater and the amount removed by the MC cooling coils, i.e.,

$$Q_{in} = Q_{EH} - Q_{cc} \quad . \tag{1.1}$$

The heat going through the center point of the specimen is one-dimensional but that which goes through the specimen near the frame has a component in the z direction as well as the y direction (see figure 2.1). The heat flux in the y direction is

$$Q_{sy} = -k_s A_s \frac{\partial T}{\partial y} \tag{1.2}$$

where the subscript "s" refers to the specimen. This is the amount of heat that would flow through the wall if it were very large so that edge effects (frame coefficients) could be ignored.

The heat flux is the amount of heat that crosses a unit surface area per unit time. Using equation (1.2), if the expression is evaluated at the geometrical center of the wall in the z direction and the front surface (y = 0), a definition of the frame coefficient

$$FCHXS = [Q_{in} - Q_{sy}(y=0,\ z=H/2)]/Q_{in} \tag{1.3}$$

If the average heat flow through the front surface of the wall is used,

$$FCMCA = 1 - \frac{w \cdot \int_0^H h_s[T_s(z,y=0) - T_{MC}]dz}{Q_{in}} \tag{1.4}$$

where w is the width of the wall. Other definitions can also be given for the flanking loss or frame coefficients as well.

2. FLANKING LOSS MODEL

2.1 THE MATHEMATICAL MODEL

FLANK calculates the temperature distribution, the heat fluxes across various boundaries and the frame coefficients. The geometry is shown in figure 2.1. Since both transient and steady-state runs are to be made in the CHB, the code must be able to be run in either mode.

All the thermophysical properties such as thermal conductivity, specific heats, etc., are assumed to be independent of temperature. The heat balance equation for the region, "A", shown in figure 2.1 is

$$\rho C_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right) , \tag{2.1}$$

where T is the temperature at any point (y, z) at time, t, ρ is the density, C_p is the heat capacity and k is the thermal conductivity. The boundary conditions are of two types. Since the CHB has coils in the MC wall to maintain a constant temperature, the constant temperature boundary condition

$$T(t) = T_o \qquad \text{on} \qquad S_2 \tag{2.2}$$

where S_2 is a line where the coils exist for heat rejection if needed.

The other type of boundary condition is a specification of the heat transfer coefficient and ambient temperature along the line S_1

$$k\frac{\partial T}{\partial y}\bigg|_{s_{iy}} + h[T_{s_{iy}}(t) - T_{AMB}(t)] = 0 \quad \text{on } S_{iy} \tag{2.3}$$

and

$$k\frac{\partial T}{\partial z}\bigg|_{s_{iz}} + h[T_{s_{iz}}(t) - T_{AMB}(t)] = 0 \quad \text{on } S_{iz} \; . \tag{2.4}$$

Here S_{iy} and S_{iz} are boundary lines along the y and z axis respectively and h is the heat transfer coeficient.

The region designated A consisting of the frame, specimens, and a portion of the MC and EC walls is shown in figure 2.2. This was broken into nodes as shown on figure 2.3. A finite difference formulation of the heat conduction equation was made in order to calculate the temperature at the node, heat fluxes across lines and the frame coefficient. Heat balances at the various nodes were put into a matrix equation formulation which resulted in the following set:

$$\underline{\underline{C}}(\underline{T}^{(P+1)} - \underline{T}^{(P)})/\Delta t + \alpha \, \underline{\underline{K}} \, \underline{T}^{(P+1)} + (1-\alpha) \, \underline{\underline{K}} \, \underline{T}^{(P)} = \alpha\underline{B}^{(P+1)} + (1-\alpha)\underline{B}^{(P)} \tag{2.5}$$

where $\underline{C}$ and $\underline{K}$ are (n x n) matrices, (n) is the number of nodes, and $\underline{T}^{(P)}$ is the temperature at the various nodes at time step "P". The $\underline{B}^{(P)}$ vector arises because of boundary conditions and α is the time integration parameter such that $0 < \alpha < 1$. If $\alpha = 1$, the solution is purely implicit (backward difference formulation), if $\alpha = 0$, it is an explicit solution scheme and if $\alpha=1/2$, the solution method is a Crank-Nicholson scheme. If $\alpha=1$ is used, no numerical instability problems will arise due to time-step sizes. The $\underline{C}$ matrix is the thermal capacity matrix and $\underline{K}$ is the thermal conductivity matrix.

Equation (2.5) can be rearranged to yield

$$\underline{C}\ \frac{(\underline{DT})^{(P)}}{\Delta t} + \alpha \underline{K}\ (DT)^{(P)} = \alpha\ B^{(P+1)} + (1-\alpha)B + \underline{K}\ \underline{T}^{(P)} \tag{2.6}$$

where

$$\underline{DT}^{(P)} = \underline{T}^{P+1} - \underline{T}^{P} \quad . \tag{2.7}$$

Equation (2.6) is the form that is actually used in FLANK. The $\underline{C}$ is a diagonal matrix where the i^{th} diagonal element is given by $\rho_i\ C_{Pi}\ \Delta y_i\ \Delta z_i$ and Δy_i and Δz_i must be evaluated for both center as well as corner nodes properly. The $\underline{K}$ matrix consists of elements for the ij node as

$$K_{ij} = k_i \frac{\Delta y_i}{\Delta z_j} + k_j \frac{\Delta z_j}{\Delta y_i} \tag{2.8}$$

and, of course, k, Δy, and Δz must be properly evaluated for each of the different types of node.

The matrix equation (2.6) is solved using three IMSL subroutines [5], LUDATF which decomposes $\underline{C}$ and $\underline{K}$ using the Crout Algorithm, LUELMF which carries out the elimination part of the solution, and LUREFF which refines the solution to achieve more accuracy.

The Crout reduction [6] or decomposition of a matrix $\underline{K}$ works basically as follows. The matrix $\underline{K}$ is written as the product of two matrices, i.e.,

$$\underline{K} = \underline{L}\ \underline{U} \tag{2.9}$$

where $\underline{L}$ is lower triangular and $\underline{U}$ is unit upper triangular, so the ij^{th} element can be expressed as

$$k_{ij} = \sum_{\ell=1}^{\min(i,j)} \lambda_{i\ell}\ u_{\ell j} \qquad (i,j = 1,2,3...n) \quad . \tag{2.10}$$

Since $\underline{U}$ is a unit upper matrix, the diagonal elements are forced to unity so $u_{ii} = 1$. Therefore,

$$k_{ij} = \lambda_{i1}\ u_{11} = \lambda_{i1} \qquad (i = 1,2,...n) \quad . \tag{2.11}$$

Also,

$$k_{ij} = \lambda_{11} u_{1j}$$

or

$$u_{1j} = k_{1j}/\lambda_{11} \qquad \text{for } j = 2,3,4,\ldots n \quad . \tag{2.12}$$

The first column of $\underline{\underline{L}}$ is determined by (2.11) and the first row of u is determined by (2.12).

After the first m-1 columns of $\underline{\underline{L}}$ and m-1 rows of $\underline{\underline{U}}$ are established, the relation

$$k_{im} = \lambda_{im} + \sum_{\ell=1}^{m-1} \lambda_{i\ell} u_{\ell m} \qquad i = m, m+1, \ldots, n$$

is used, or solving for λ_{im},

$$\lambda_{im} = k_{im} - \sum_{\ell=1}^{m-1} \lambda_{i\ell} u_{\ell m} \tag{2.13}$$

The relationship which permits the computation of u_{mj} is

$$u_{mj} = \frac{1}{\lambda_{mm}} [k_{mj} - \sum_{\ell=1}^{m-1} \lambda_{m\ell} u_{\ell j} \tag{2.14}$$

Notice that the m^{th} row of $\underline{\underline{U}}$ and the m^{th} column of $\underline{\underline{L}}$ are determined by eqs. (2.14) and (2.13), respectively.

The Crout algorithm requires about $n^3/3$ multiplications to accomplish this decomposition so that for a 246 x 246 matrix about 5 million multiplications are required. If an error message IER = 129 is printed by FLANK, this came from LUDATF indicating that the matrix is "algorithmically singular" indicating a small pivot element was encountered in the decomposition process. If the message IER = 34 is printed, the computed solution is in error by more than can be accounted for by the uncertainty of the data.

LUDATF, thus, triangularizes the matrix $\underline{\underline{K}}$ into $\underline{\underline{L}}$ and $\underline{\underline{U}}$. The decomposition is always possible provided the leading principal minor determinants are all nonzero, i.e.

$$\det k_{11} \neq 0 \quad ,$$

$$\begin{vmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{vmatrix} \neq 0$$

$$\vdots$$

$$\det \underline{\underline{K}} \neq 0$$

This triangular decomposition is unique if the diagonal elements of either $\underline{\underline{L}}$ or $\underline{\underline{U}}$ are specified.

To obtain a first estimate of the temperature vector $\underline{T}_1$, set

$$\underline{\underline{L}}\ \underline{\underline{U}}\ \underline{T}_1 = \underline{B}$$

and then

$$\underline{\underline{L}}\ \underline{X} = \underline{B} \quad .$$

$\underline{X}$ can be solved directly by forward substitution by noting

$$X_i = (b_i - \sum_{j-1}^{i-1} \lambda_{ij} X_j)/\lambda_{ii} \quad .$$

The temperatures are then obtained by back substitution of the matrix equation

$$\underline{\underline{U}}\ \underline{T}_1 = \underline{X} \quad .$$

Once the decomposition has been accomplished, LUELMF calculates the temperatures, so that a first estimate $\underline{T}_1$ is obtained as indicated. LUREFF is then called to refine the solution. This routine basically works by calculating a residual $\underline{r}_1$ from

$$\underline{r}_1 = \underline{b} - \underline{\underline{K}}\ T_1 \quad . \tag{2.15}$$

The new solution $\underline{y}_1$ is obtained (using the same decomposition from LUDATF) from

$$\underline{\underline{K}}\ \underline{y}_1 = \underline{r}_1 \tag{2.16}$$

which is then added to $\underline{T}_1$ to obtain a better estimate $\underline{T}_2$, i.e.,

$$\underline{T}_2 = \underline{T}_1 + \underline{y}_1 \quad . \tag{2.17}$$

This computation requires about $2n^2$ multiplications. This procedure is continued by now obtaining

$$\underline{r}_2 = \underline{b} - \underline{\underline{K}}\, T_2 \quad , \tag{2.18}$$

computing a $\underline{y}_2$, obtaining a $\underline{\underline{T}}_3$, etc.

Basically, the procedure continues until the accuracy specified by IDGT is satisfied or an error is produced. Again IER = 129 means the matrix is too ill-conditioned to achieve the desired accuracy. For example, if IDGT is set to three, an accuracy test is performed to be sure that when $\underline{\underline{K}}\underline{T}$ is calculated, the first three digits of each element is the same as the first three digits of $\underline{b}$. If IDGT = 0, this test is bypassed.

2.1.1 Heat Transfer Coefficients

The heat transfer coefficient must be read into the code. The CHB permit various air velocities and temperature conditions to be imposed on the specimen. The range of heat transfer coefficients is between 1.0 to 25 $W/m^2 \cdot K$. The heat transfer correlation used for laminar flow is [7]

$$\bar{h} = 0.664 \sqrt{Re}\, Pr^{1/3} (k/L), \tag{2.19}$$

where

Re = the Reynolds number,
Pr = the Prandtl number,
k = the thermal conductivity of air ($W/m \cdot K$)
L = the length of the wall in the direction that the air is moving,

and

$\bar{h}$ is the average convective heat transfer coefficient ($W/m^2 \cdot K$).

This expression is used if $Re < 5 \times 10^5$.

If the Reynolds number is greater than 5×10^5, the flow becomes turbulent and there are mixed boundary layer conditions in that for some of the wall specimen the flow is laminar and turbulent flow dominates for the remainder of the wall. In this case

$$\bar{h}_L = \frac{1}{L} \left[\int_0^{X_c} h_{laminar}\, dx + \int_{X_c}^{L} h_{turbulent}\, dx\right] \tag{2.20}$$

and it is necessary to calculate the transition distance X_c. The result of the integration is

$$\bar{h}_L = \left(\frac{k}{L}\right) [0.037\, Re^{4/5} - 836] Pr^{1/3} \tag{2.21}$$

Figure 2.4 is a plot of the forced convection heat transfer coefficients. These will apply to both the MC and EC.

If the air velocity is very low, then natural convection becomes important and even though there is slight forced air motion, both types of convection must be considered.

Air infiltration is not taken into account in the heat diffusion equation. The FRP is essentially impermeable to air and moisture but the wall specimen may not be. FLANK would require extensive modifications for the specimen region in order to take into account infiltration since convection may be an important aspect of the heat transport mechanisms.

2.2 STEADY-STATE ANALYSIS

FLANK can be used either for steady-state calculations or dynamic calculations. From equation (2.5) it is seen that if α is set to 1 and Δt is very large (10^{30}), then the static equations for heat transfer result. In FLANK, if ITRAN = 0, these steps are automatically performed.

The boundary conditions are of the convective type or else the temperature may be specified at various boundary points.

The equation that is solved in the steady-state case is

$$\underline{\underline{K}}\,\underline{T} = \underline{B} \tag{2.22}$$

where $\underline{B}$ arises because of the boundary conditions imposed.

The output from the static calculations consists of:
a) echo of the input data;
b) the temperature at each of the nodes;
c) the heat transfer across various surfaces; and
d) the frame coefficients.

2.3 TRANSIENT ANALYSIS

If the control parameter ITRAN is set to unity, a dynamic calculation will be made. The initial temperatures and the boundary conditions imposed must be part of the input file. In this case equation (2.6) is rewritten as

$$[\frac{1}{\Delta t}\,\underline{\underline{C}} + \alpha\underline{\underline{K}}](D\underline{T})^P = \alpha\underline{B}^{(P+1)} + (1 - \alpha)\underline{B}^{(P)} - \underline{\underline{K}}\,\underline{T}^P \quad , \tag{2.23}$$

or

$$\underline{\underline{A}}(D\underline{T})^P = \underline{C}^P \quad . \tag{2.24}$$

The boundary conditions may be changing in time so that $\underline{B}$ may be a time-dependent quantity. Equation (2.24) is solved for $\overline{(D\underline{T})}^P$ and then the actual temperatures are obtained at time (P + 1) by using the relation

$$\underline{T}^{(P+1)} = \underline{T}^{(P)} + (D\underline{T})^P \quad . \tag{2.25}$$

The output of FLANK is as follows: a) echo of input data; b) the temperature at each of the nodes at each time step; c) the heat transfer across various surfaces at each time step; and d) the frame coefficients at each time step.

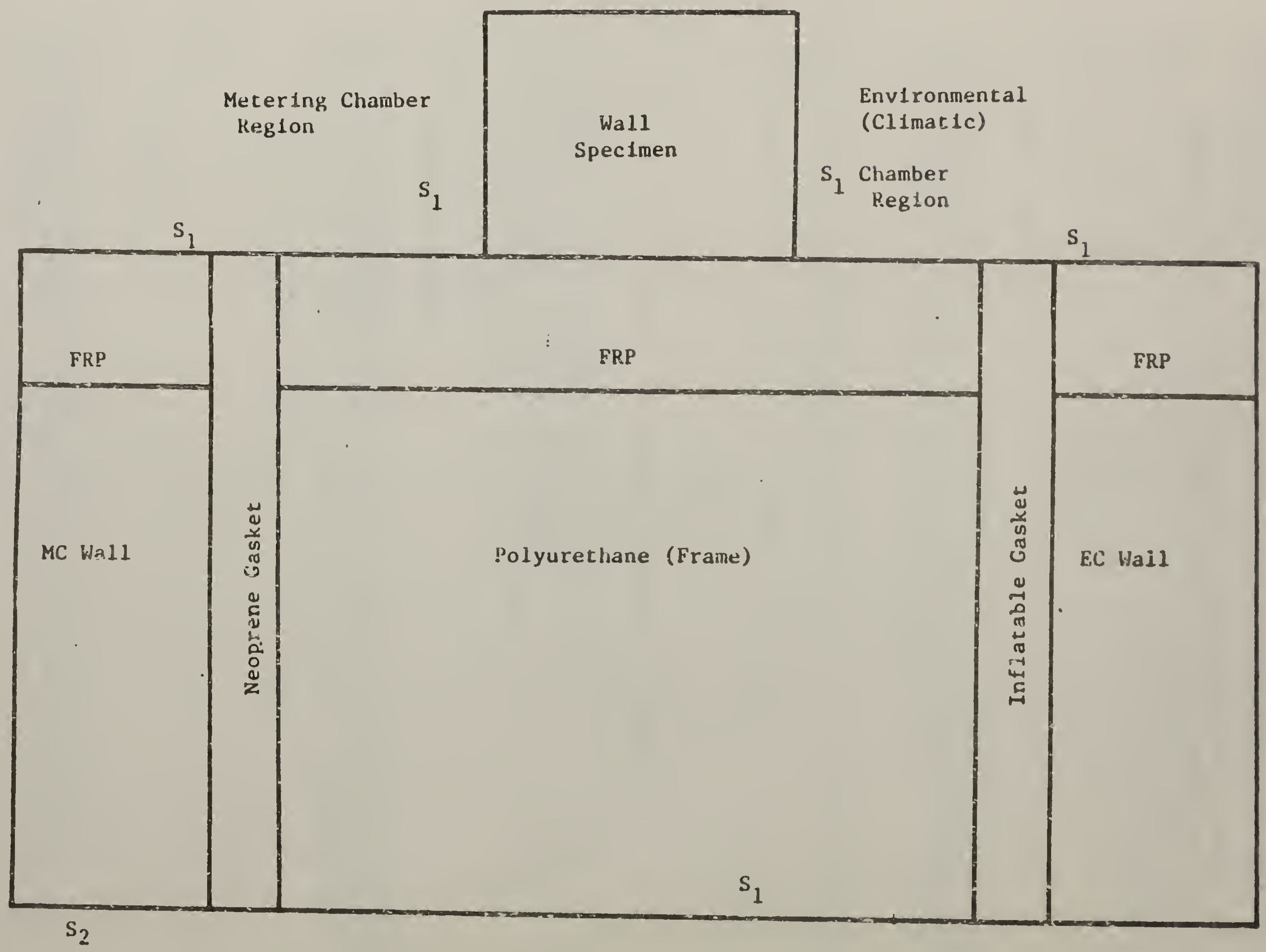

Figure 2.1 Geometry (Region A) of the CHB frame, wall specimen, gaskets, and portions of the MC and EC walls

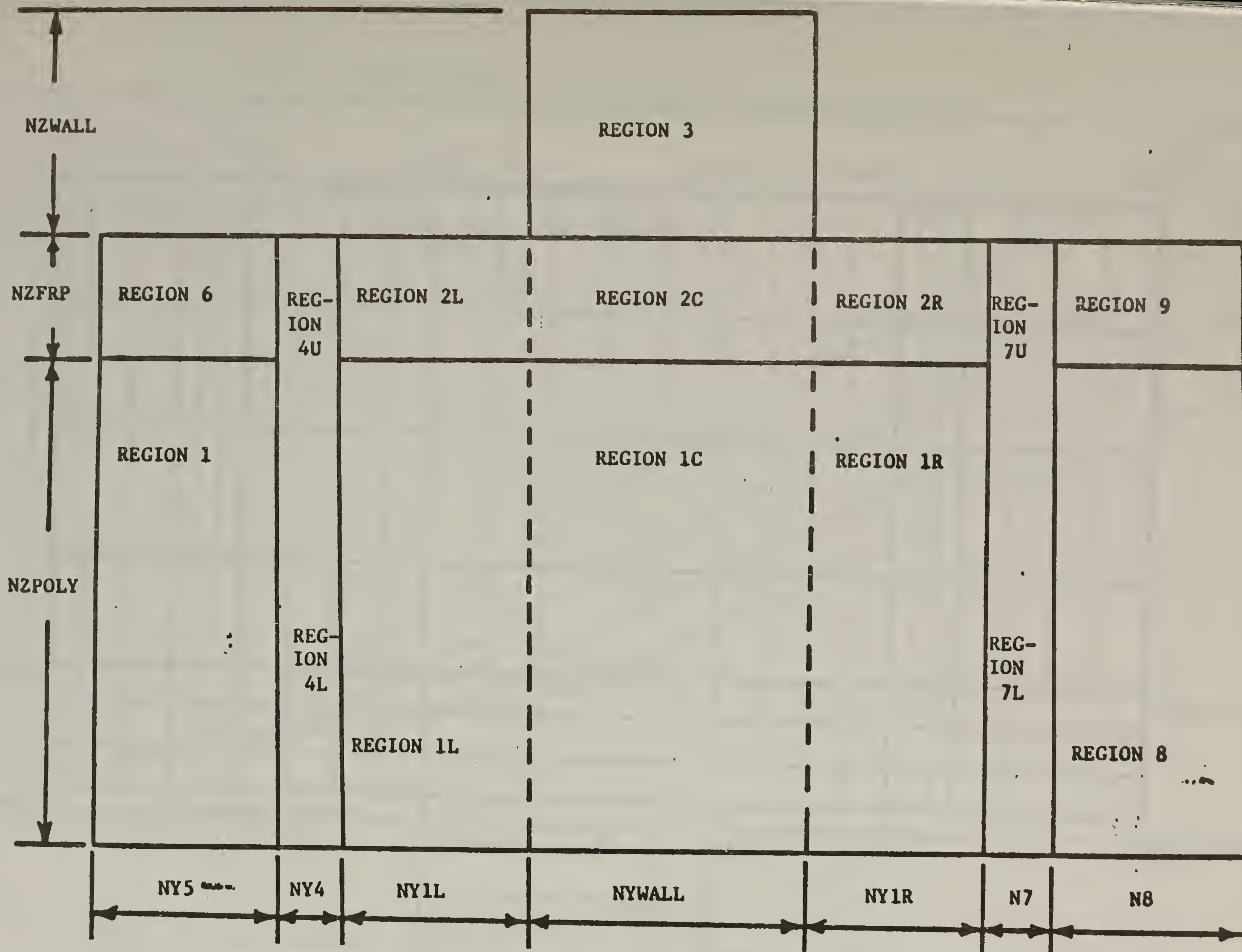

Figure 2.2 The various regions and the number of nodes in each region for the CHB as coded in FLANK

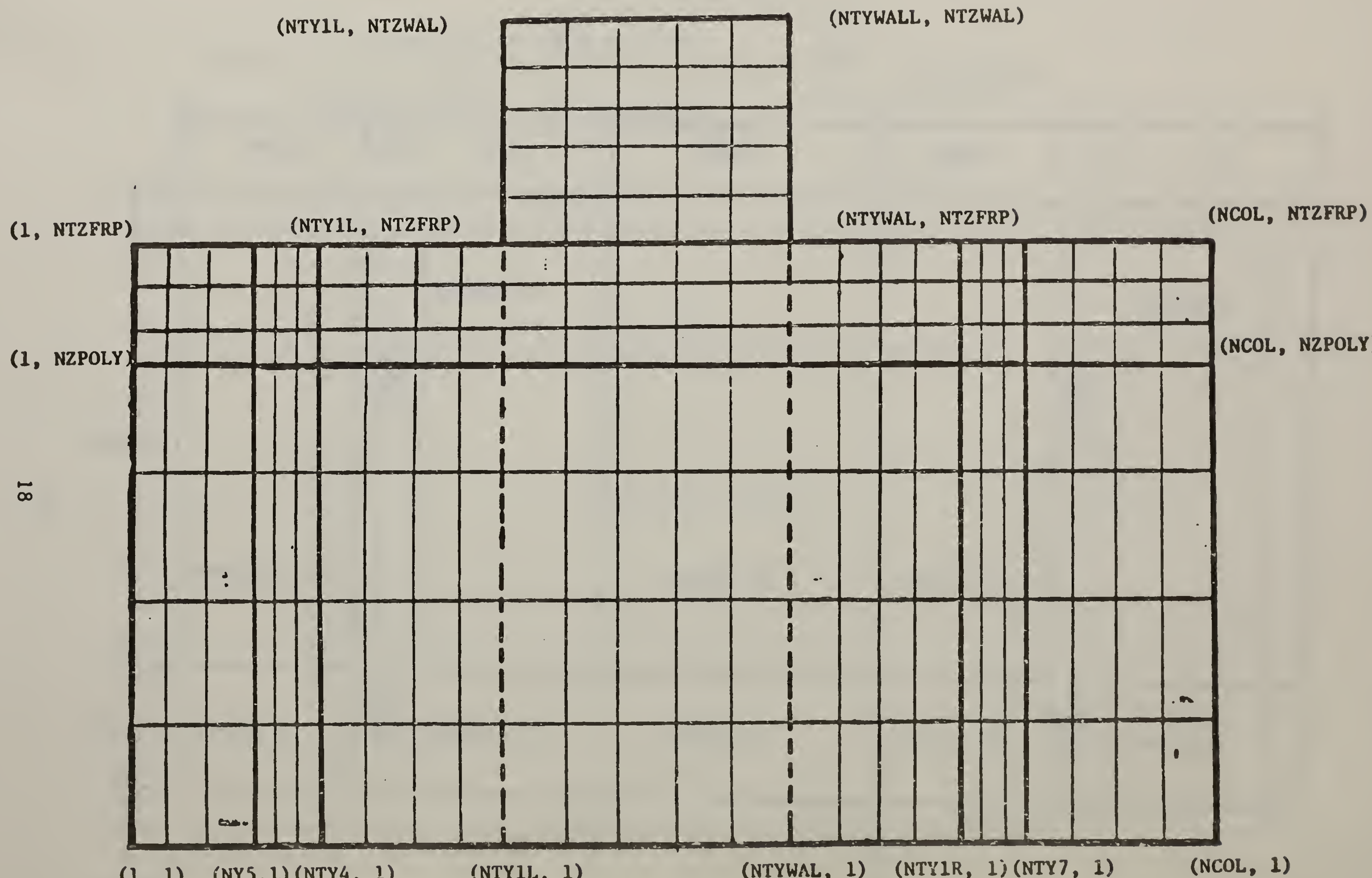

Figure 2.3 Nodal structure of the frame, specimen, gaskets and portions of the MC and EC walls

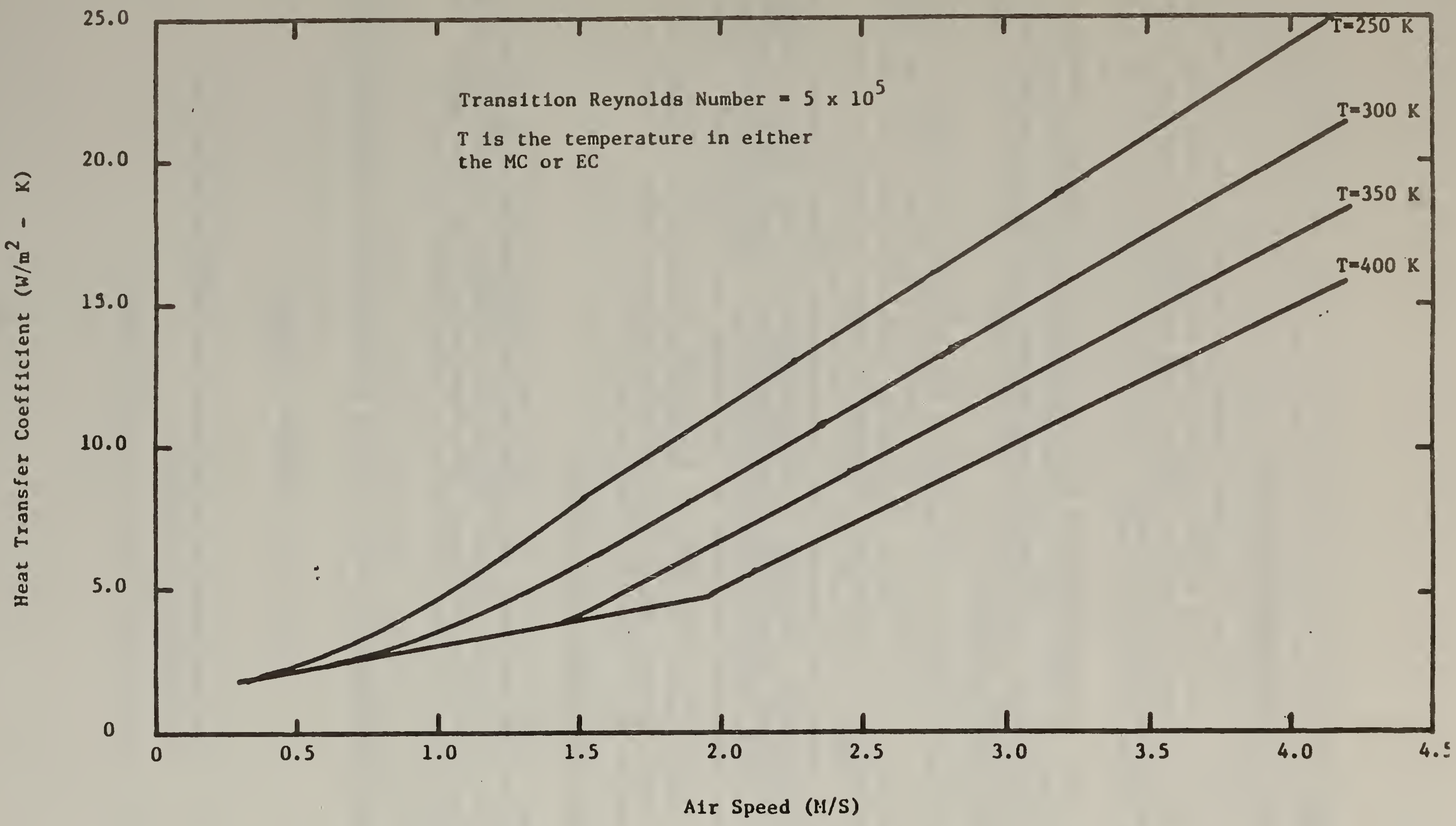

Figure 2.4 Heat transfer coefficient due to forced convection that applies to the metering and environmental chamber specimen and frame surfaces

3. RESULTS

3.1 CODE VERIFICATION AND DISCUSSION

FLANK was run for various situations to obtain an indication that it is working properly. For example, if all the temperatures imposed on all boundaries are equal, then the frame and specimen must go from the initial temperature to the temperature specified by the boundary conditions. This test was done and the program provided the correct answers.

Various other tests were done such as varying the node structure to determine the "minimum acceptable" number, looking at the time steps and the accuracy of the calculation, etc. Additionally, heat balances for given regions are performed to determine whether the heat entering a region is equal to that leaving in a steady-state calculation.

The node structure used in calculating all the results in this report is shown in figure 3.1. The accuracy of the calculation as a function of the node size is shown in figure 3.2. Figure 3.3 gives the surfaces for which the heat fluxes are calculated. It is found that 240 nodes or greater (if they are chosen reasonably) will provide good results. Of course, the more nodes chosen the greater the memory requirements. The time step used provides different accuracy for different nodes. Generally, if the temperature of a node does not change rapidly (as in the center of the polyurethane frame), longer time steps can be used. Figure 3.4 shows the effect of the time step for node 121 which is one of the most sensitive nodes since the FRP changes temperature rapidly. These results are essentially the same for $\alpha = 1/2$ and $\alpha = 1.0$. This means that for highly accurate results in the FRP, time steps of the order of 5 minutes or so are required. However, the accuracy in the polyurethane region is good even for time steps of the order of an hour or so.

3.2 STEADY-STATE RESULTS AND ANALYSIS

The frame coefficient correction to the total energy balance of the CHB will generally be less than about ten percent of the total heat entering or leaving the MC side of the CHB. However, this is not necessarily the case since as T_{MC} approaches T_{EC} with T_{AMB} different, then essentially no heat flows through the wall specimen and the frame coefficient can approach unity as shown in figure 3.5.

An overall heat balance on the MC is given by the relation (assuming $T_{MC} > T_{EC} > T_{AMB}$

$$Q_{in} = Q_{out} = Q_{specimen} + Q_{frame} \tag{3.1}$$

where

Q_{in} = The watts supplied by the heater and the blower minus the watts taken by the chiller,

Q_{out} = The heat that leaves the metering chamber.

It is assumed that the heat lost through the rest of the metering chamber walls is negligible since they are maintained at a constant temperature by coils on the outside. If this is not the case, another term $Q_{MCwalls}$ must be added to equation (3.1). The heat loss through the MC walls is given by the relation

$$Q_{MCwalls} = k_{MC} A_{eff} \frac{T_i - T_{AMB}}{L_{MC}} \tag{3.2}$$

where k_{MC} = the effective thermal conductivity of the MC walls (W/m-K)
L_{MC} = the thickness of the MC walls, m
T_{AMB} = the temperature on the outside of the walls
A_{eff} = the effective area normal to the heat flow (m^2).

It is assumed that the surface resistances are negligible compared to the resistance of the walls in equation (3.2). Using conduction shape factors to determine A_{eff} [8], the shape factors for the edge of two adjoining walls is 0.54 x (length of the edge) and for a corner, the shape factor is 0.15 x (the thickness of the wall). Therefore,

$$A_{eff} = \sum_{i=1}^{5} A_{in,i} + 0.54\, L_{MC} \cdot \sum_{i=1}^{8} e_i + 4 \cdot (0.15) L_{MC} \tag{3.3}$$

where e_i = length of edge i (m) and the 4 in equation (3.3) is because there are four corners in the MC. The inside dimensions of the NBS-CHB MC are 1.111 m x 4.572 m. Therefore,

$$A_{eff} = 13.94 + 2(5.079) + 2(3.386) + 0.54(0.4572) \cdot [2 \times 4.572$$
$$+ 2 \times 3.048 + 4 \times 1.11] + 0.60(0.4572) = 36.00\ m^2 \quad .$$

Assuming that there is a 1 °C difference between the inside and the outside of the MC box,

$$Q_{MCwalls} = 0.022 \frac{W}{m \cdot {}^\circ C} (36\ m^2) \times \frac{1(^\circ C)}{0.457(m)}$$
$$= 1.73 \text{ watts}$$

Going back to equation (3.1), assuming temperatures T_{MC1} and T_{EC1} on the specimen surfaces (for a uniform temperature),

$$Q_{frame} = Q_{in} - Q_{specimen}$$
$$= Q_{in}[1 - \frac{Q_{specimen}}{Q_{in}}] \tag{3.4}$$
$$= Q_{in} \cdot FCHXS,$$

if

$$Q_{specimen} = k_s A_s (T_{EC1} - T_{MC1})/\Delta Y_s \quad , \tag{3.5}$$

or

$$Q_{frame} = Q_{in} \cdot FCMCA \quad , \tag{3.6}$$

if

$$Q_{specimen} = 4.572 \int_{z_{FRP}}^{z_{wall}} [T(z) - T_{MC}]\, h(z)\, dz \quad , \tag{3.7}$$

where the 4.572 is the width of the wall specimen in meters. T(z) is the surface temperature of the wall as a function of height on the MC side of the specimen.

If $Q_{MCwalls}$ is not much smaller than Q_{frame}, then eqs. (3.4) and (3.6) must be modified to take that into account, i.e.,

$$Q_{frame} = Q_{in} - Q_{specimen} - Q_{MCwalls}$$

$$= Q_{in} \left(1 - \frac{Q_{specimen}}{Q_{in}}\right) - Q_{MCwalls} \; . \tag{3.8}$$

The program FLANK computes the frame coefficients FCHXS and FCMCA. Various steady-state runs were made to obtain the temperature distribution, heat fluxes, and frame coefficients. All runs were made using 246 nodes. Figure 3.6 illustrates the variation of the frame coefficients with the thermal conductivity of the wall specimen. In this case, the specimen was placed 0.127 m (5 in) from the MC side of the frame.

Figure 3.7 shows the influence of the position of the specimen on the wall frame. The thermal conductivity of the wall specimen was taken to be 0.036 W/m•°K for this series of runs. The frame was 0.3048 m (12 in) wide. The frame coefficients increase with position from the frame edge as expected. The thickness of the specimen is 0.1016 m (4 in).

The thickness of the wall specimen also will influence the frame coefficient. Figure 3.8 shows FCHXS and FCMCA both increase with wall thickness as expected. The 0.3048 m wide (12 in) frame was used. It appears that FCMCA is starting to bend toward a less than linear increase with thickness.

The frame coefficient as defined in this report is, strictly speaking, a steady-state concept even though it will be examined in time as well. If the boundary conditions vary in time or if the system is not initially in equilibrium, the transient or a dynamic analysis must be performed.

3.3 TRANSIENT ANALYSIS AND RESULTS

A transient analysis differs from a dynamic analysis in that the latter has driven or time-dependent boundary conditions. A transient analysis examines the temperature response due to the fact that the system does not start out in an equilibrium temperature distribution. The system approaches the steady-state asymptotically so that "time constants" of the system can be obtained. A series of runs with this objective were made.

Figure 3.1 shows the node numbering system used for all the transient calculations shown. Figure 3.9 shows the temperature at nodes 68 and 226 as a function of time. The initial temperatures were all assumed to be 20°C and at time zero, an MC temperature of 38° and a room temperature of 25°C was impressed on the system. The climatic chamber was taken to be 20°C also. These boundary conditions were held constant in time and the equilibrium temperature approached as shown. Note that in the specimen (made of polystyrene) the temperature reaches equilibrium in about two or three hours whereas deep in the polyurethane, it takes about 50 hours to reach equilibrium.

If one examines the thermal diffusivities, α, one can see the reason for these times. The thermal diffusivity of the polyurethane is $k/\rho c$ (see table 1.6) which is $(0.022\ \mathrm{w/m \cdot K})/(32\ \mathrm{kg/m^3})\ (921\ \mathrm{J/kg \cdot K})$ or $7.5 \times 10^{-7}\ \mathrm{m^2/s}$ and $0.036/(27.64)(1214)$ or $1.08 \times 10^{-6}\ \mathrm{m^2/s}$ is the thermal diffusivity for the beaded polystyrene. The Fourier number is a dimensionless time and is defined as

$$Fo \equiv \alpha t/L_C^2 \tag{3.9}$$

where L_C is the characteristic length of the system. The time dependence can roughly be related to e^{-BiFo} where Bi is the Biot number ($= h\ L_C/k$). For the frame $Bi \simeq 2.33\ (0.04)/0.022 = 4.2$ using the thermal conductivity of the polyurethane instead of the FRP. The time constant ("1/e" value) is thus

$$Fo \cdot Bi = 1 \tag{3.10}$$

or for $L_C = 0.229$ m, the time to reach about 60-70 percent of the steady-state value is 5 hours. This is a very crude estimate but even without using FLANK, it can be seen that time lags of the order of tens of hours will be required for the polyurethane to reach equilibrium.

The approach of the frame coefficients to their equilibrium value is shown in figure 3.10. In this case, the frame and specimen were initially at 20°C and suddenly the metering chamber temperature was switched to 38°C and the ambient (room) temperature was at 25°C. The time behavior of the frame coefficients is shown to decrease to their equilibrium (steady-state) values in about 40 hours. Initially a large amount of heat is required to heat the frame material itself and then, after steady-state temperatures are achieved, the frame coefficient approaches the steady-state value.

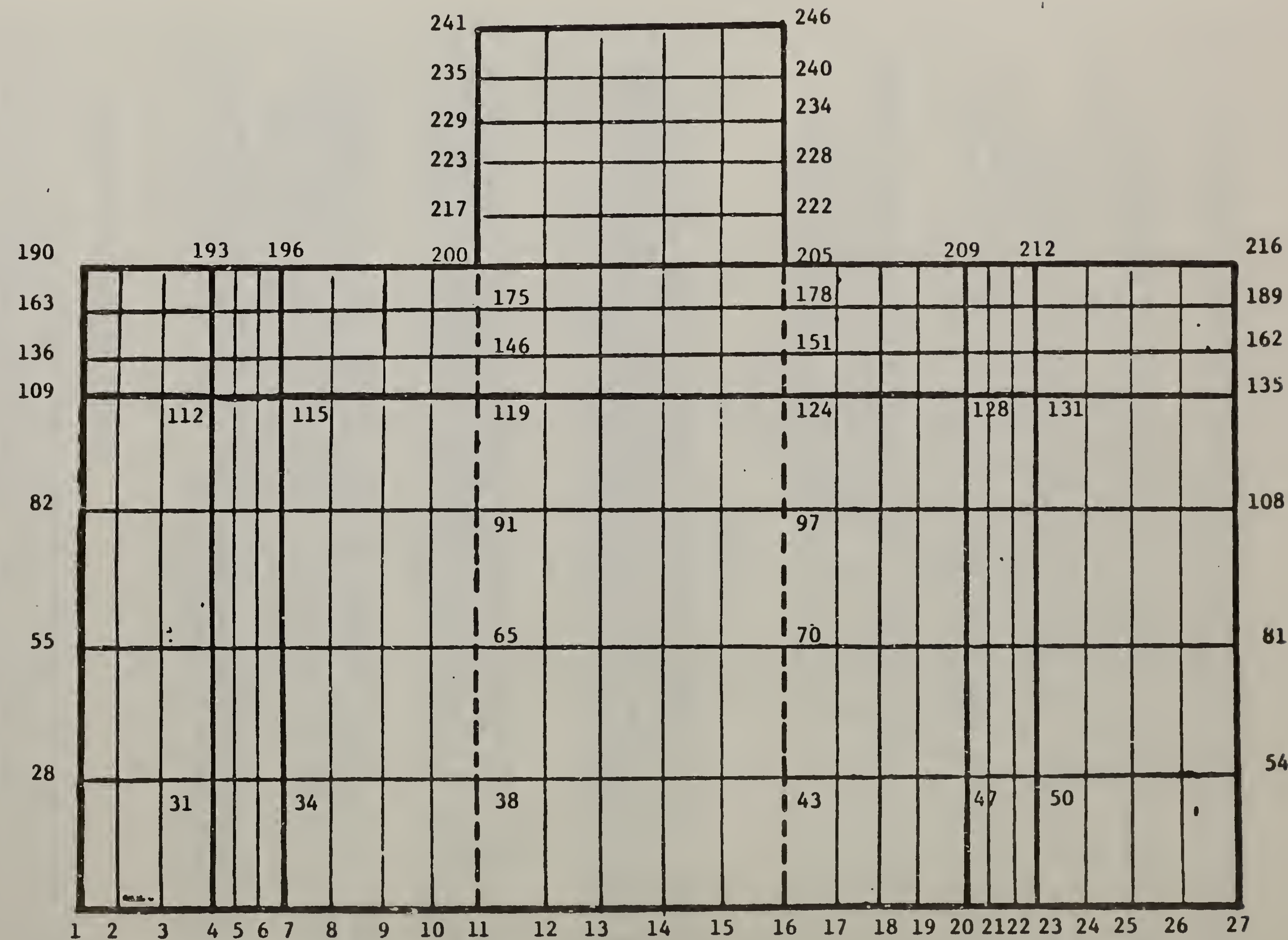

Figure 3.1 The node structure used in the calculations presented

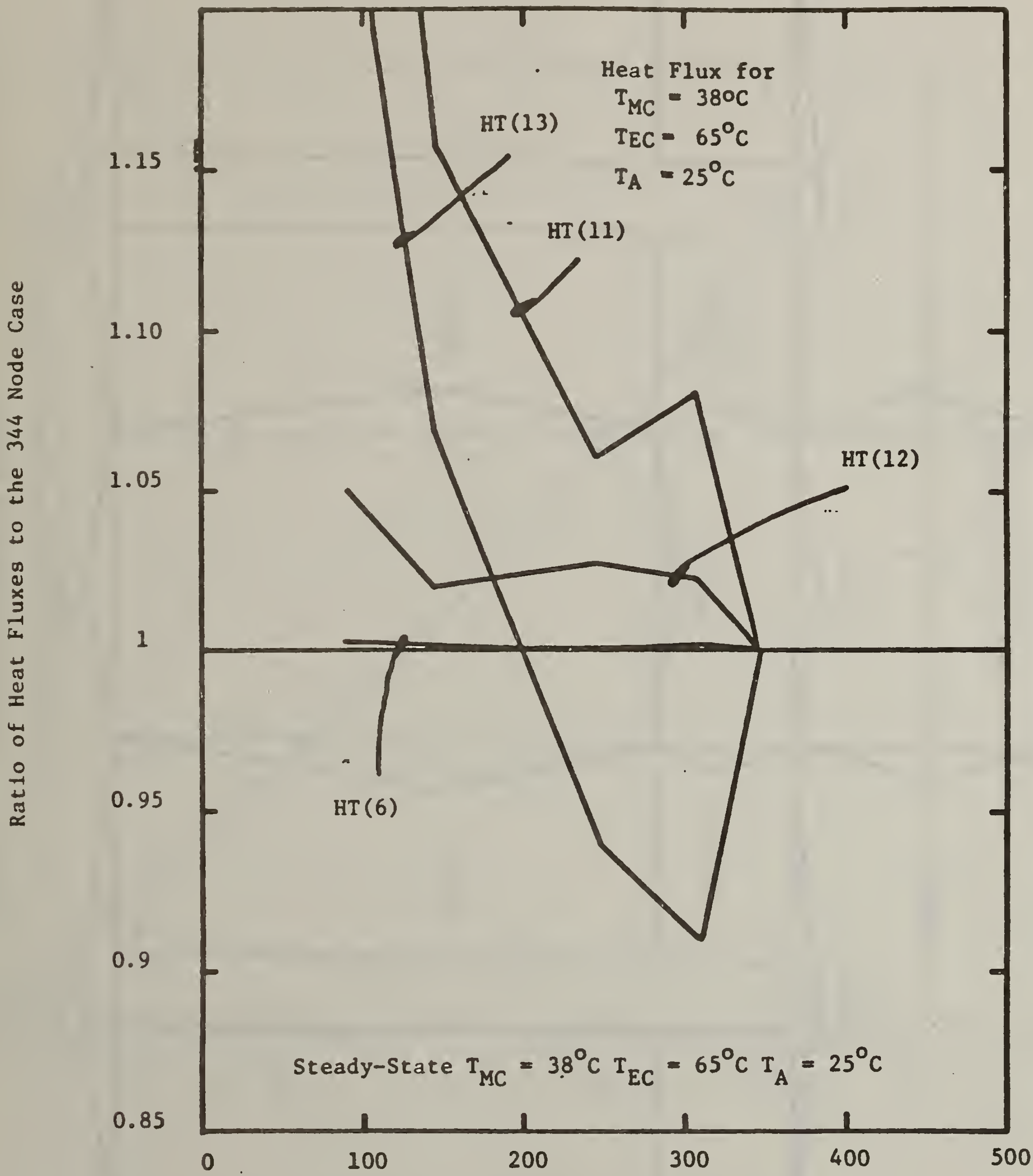

Figure 3.2 The influence of node structure on the accuracy of heat fluxes across various surfaces

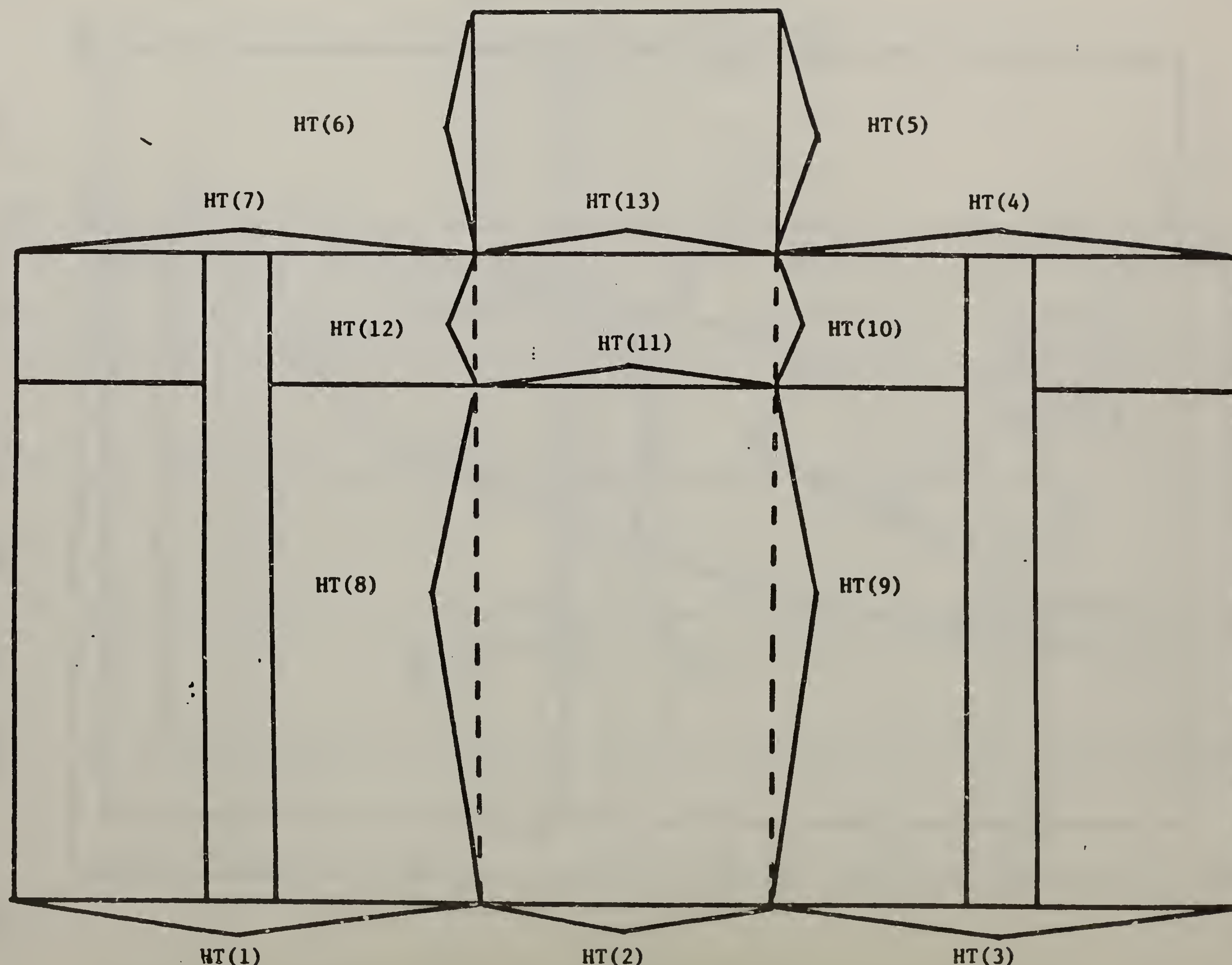

Figure 3.3 Heat fluxes numbering system

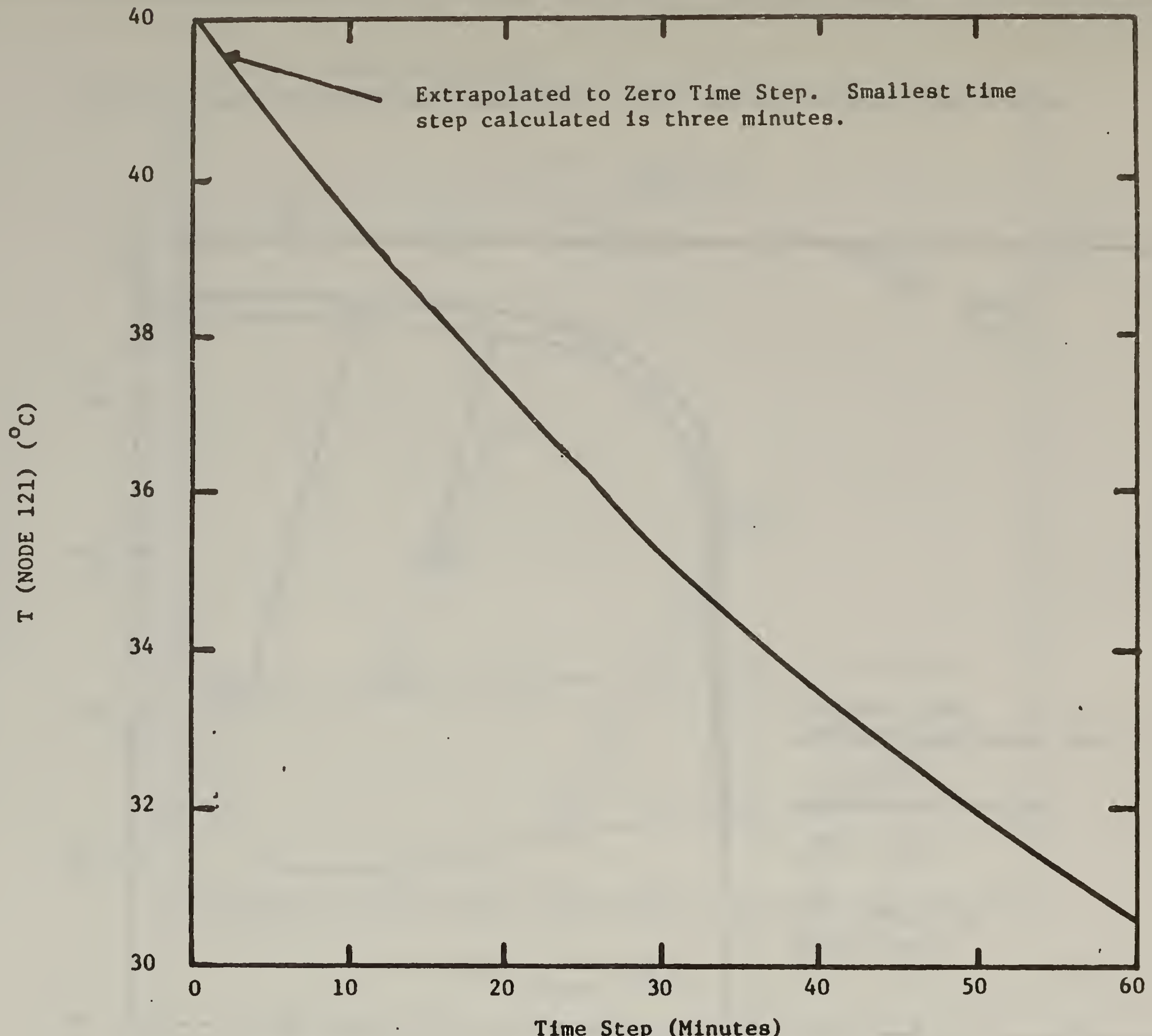

Figure 3.4 Sensitivity of the temperature in the FRP to the time step

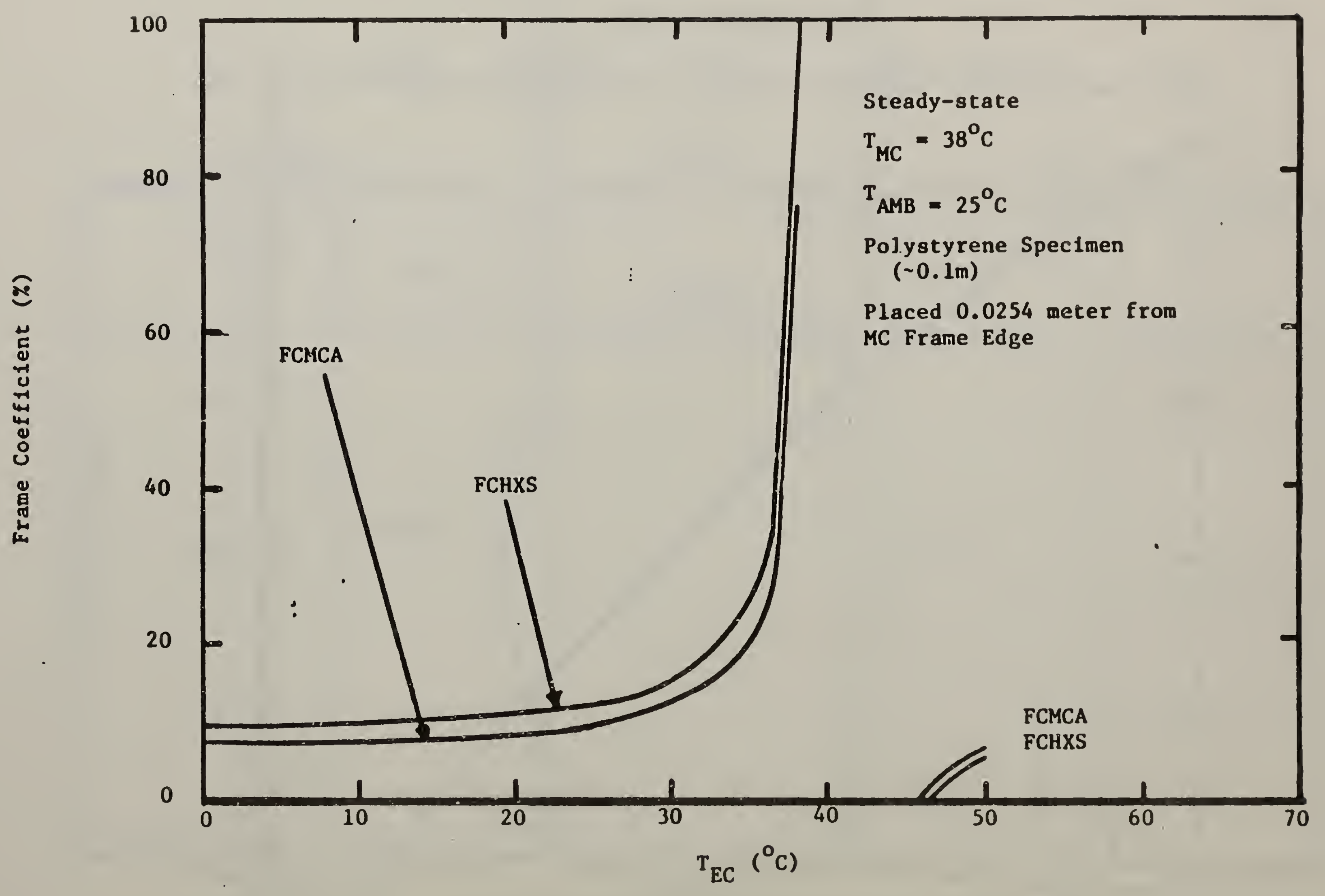

Figure 3.5 The influence of a variation of the climatic chamber temperatures on the frame coefficient

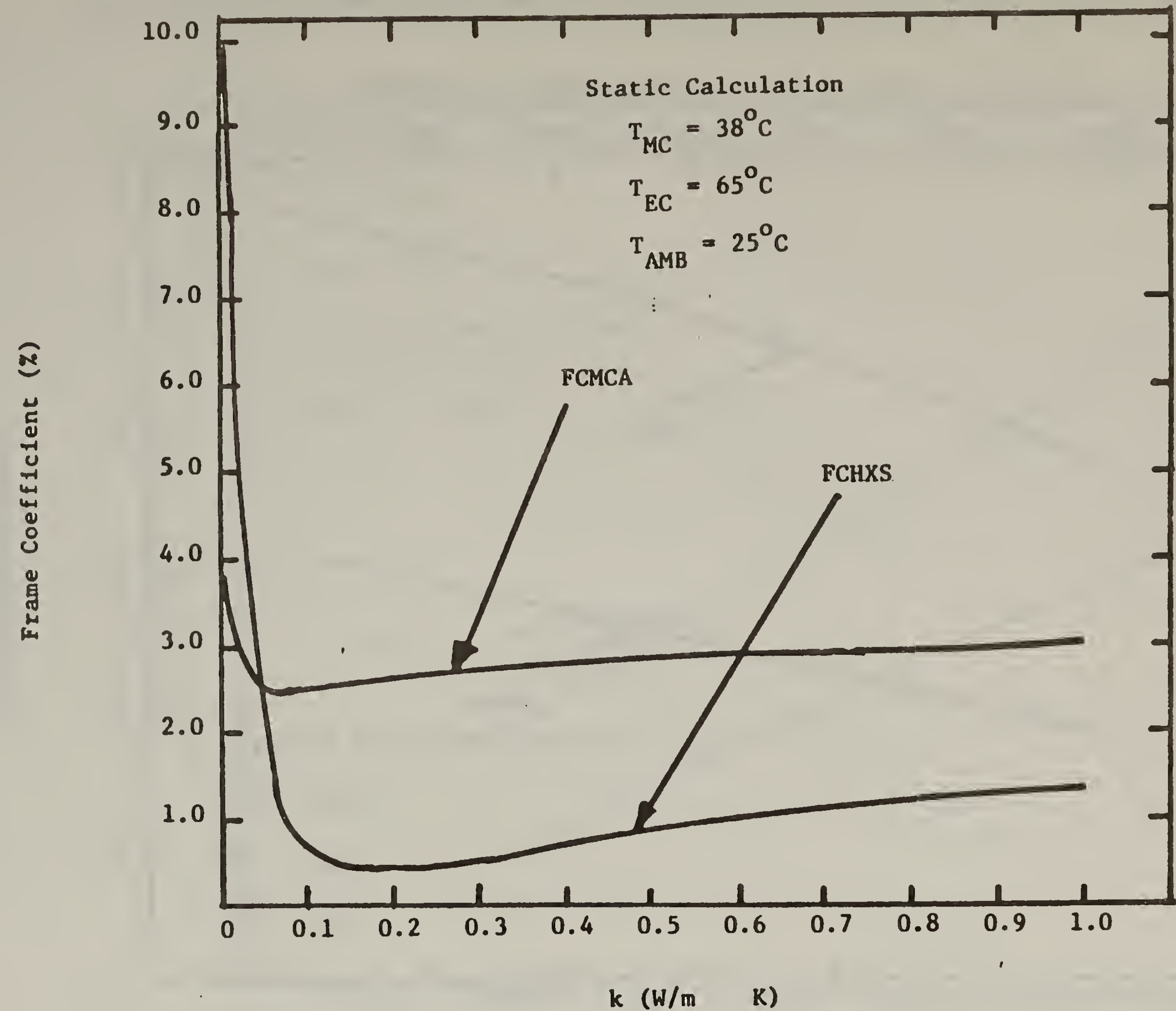

Figure 3.6 The variation of the frame coefficients with the thermal conductivity of the wall specimen

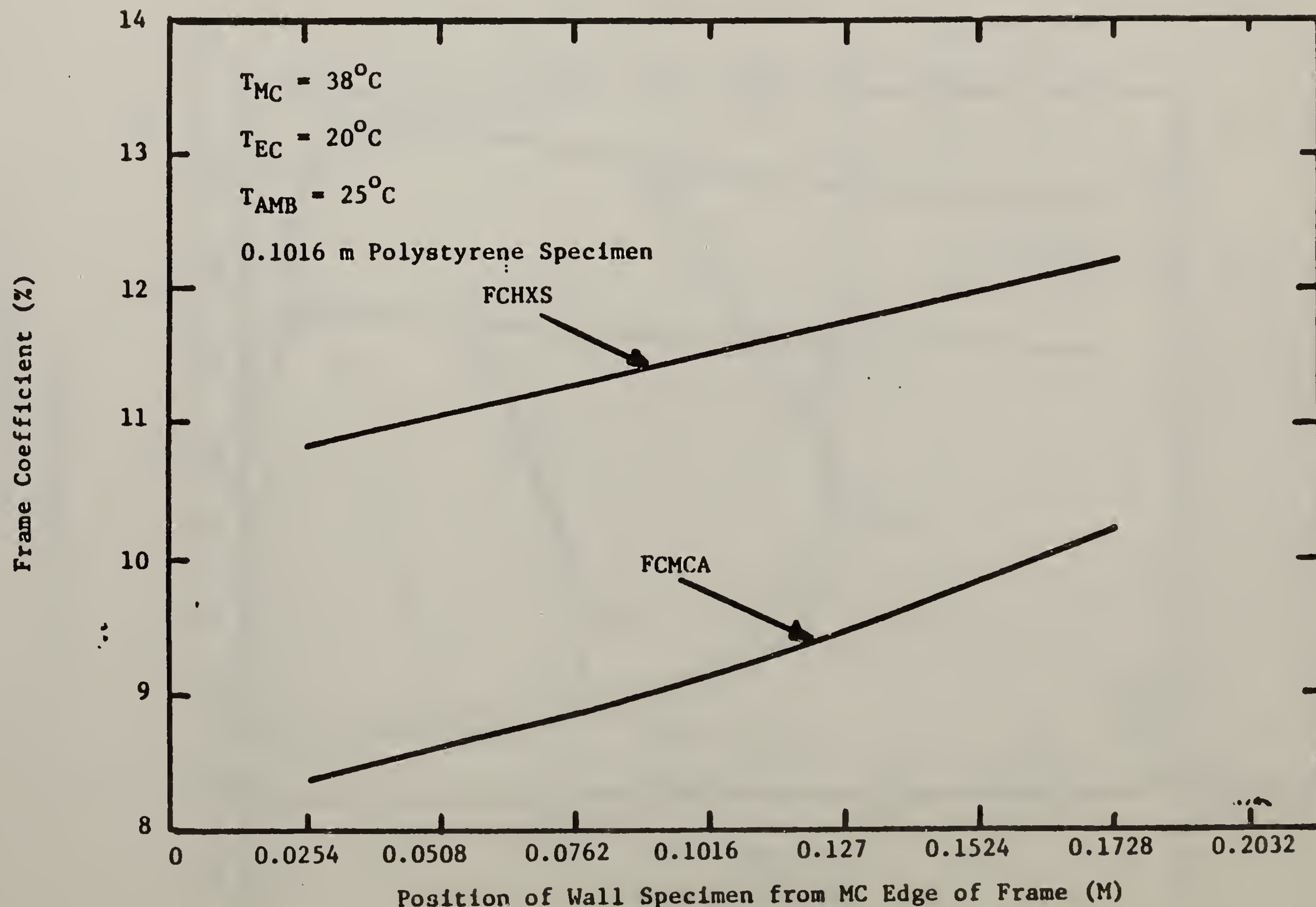

Figure 3.7 The frame coefficient as a function of position from the metering chamber face of the 0.3048 m wide frame

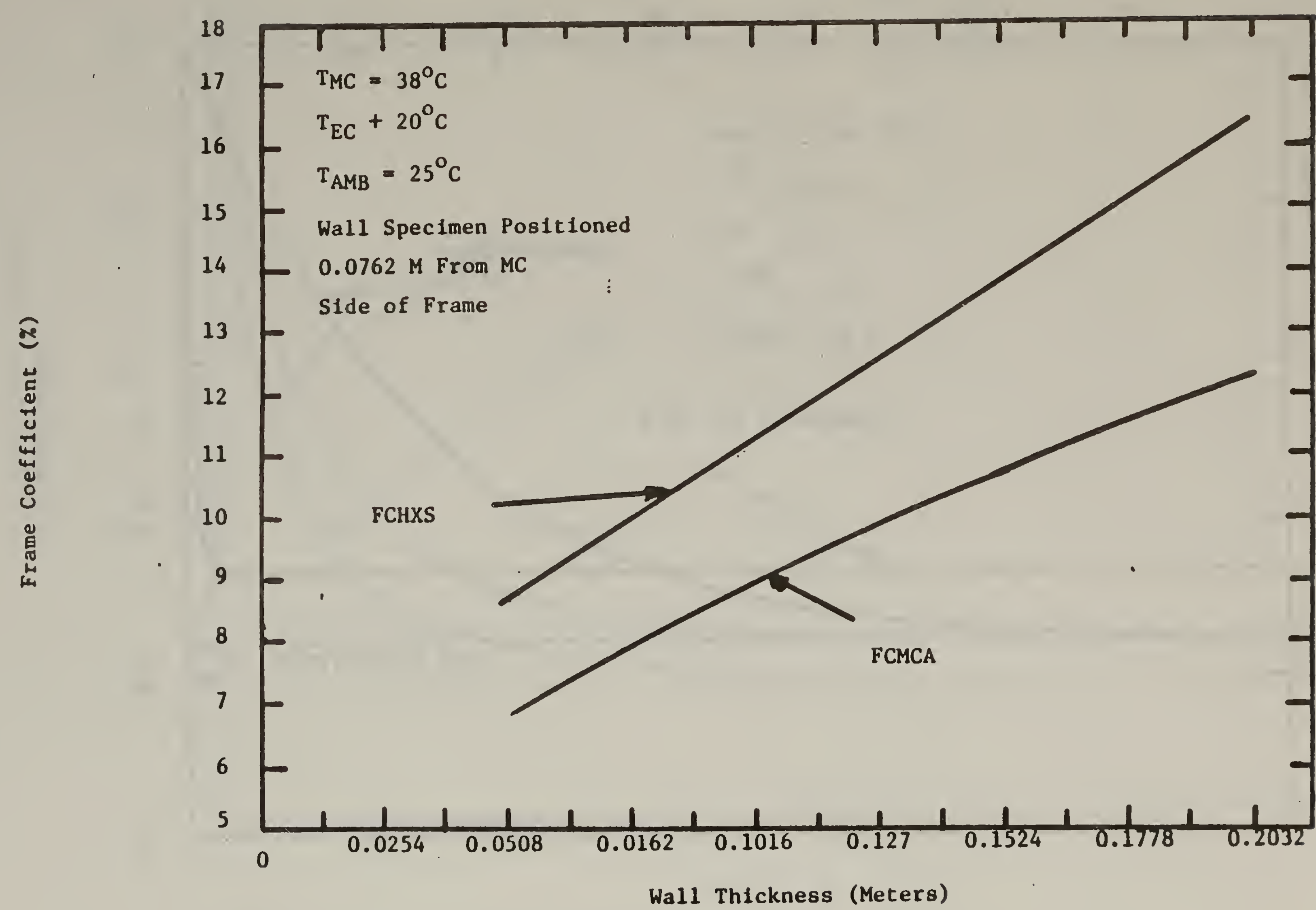

Figure 3.8 The variation of the frame coefficient with the wall thickness

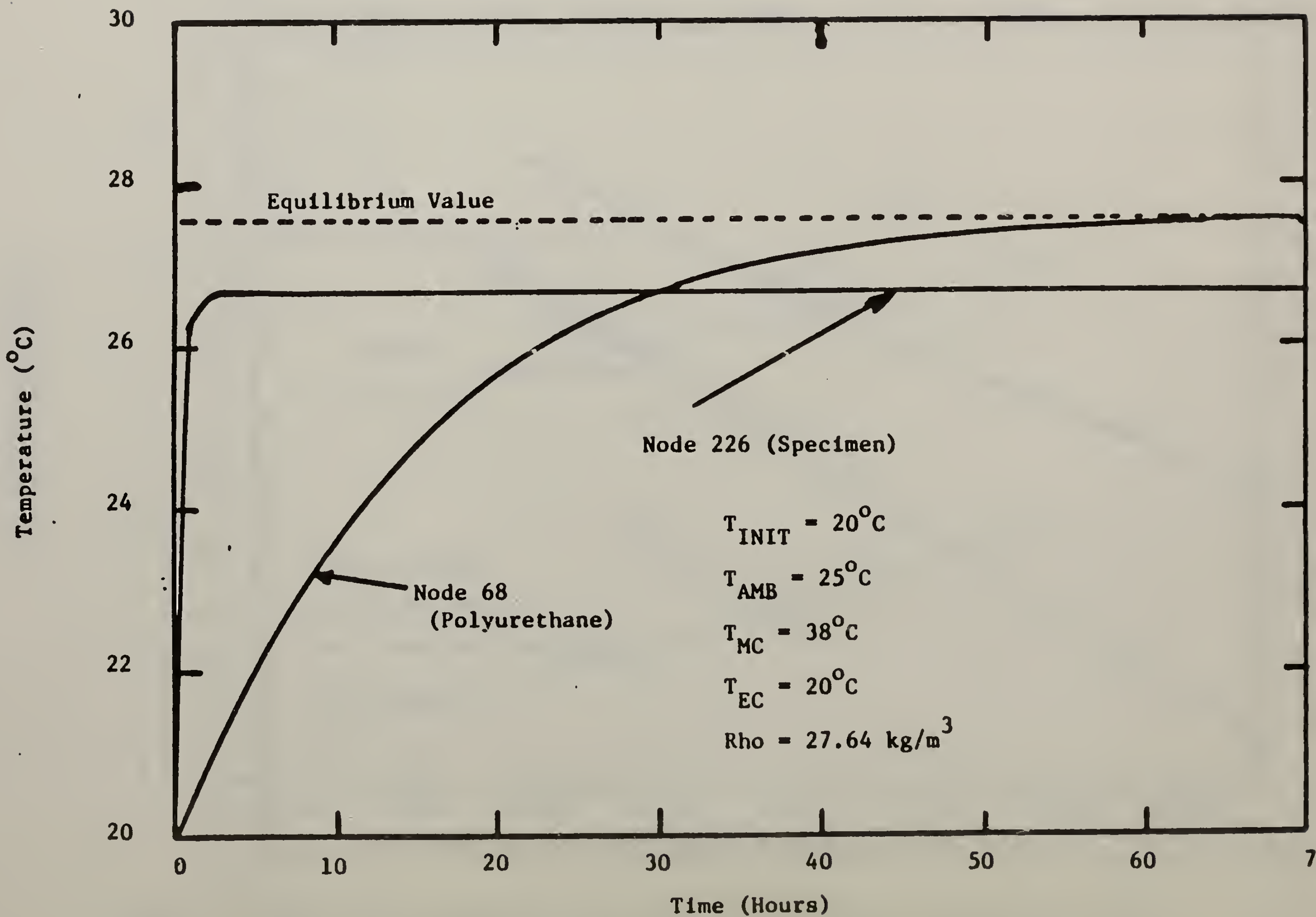

Figure 3.9 The transient response of the polyurethane frame and the specimen to abrupt changes in boundary conditions

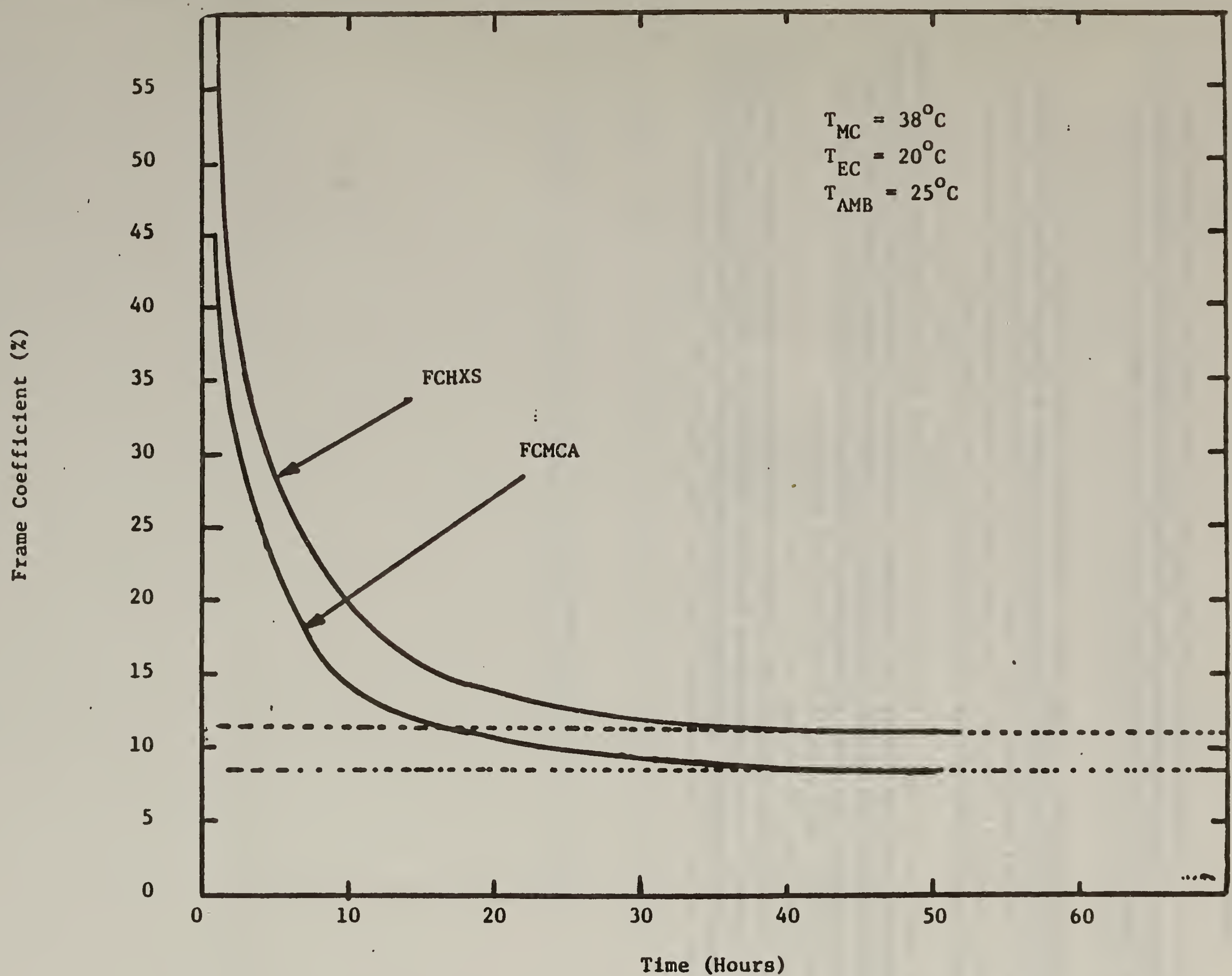

Figure 3.10 The transient response of the frame coefficients

4. SUMMARY

An initial analysis of the NBS-CHB has been done using the code FLANK (see Appendix). The temperature distribution, heat fluxes across various boundaries, and frame coefficients have been calculated for steady-state and transient situations.

The time constants for the frame coefficients are of the order of 40 hours. The heat transfer coefficients are held constant during all transient analysis but it is believed that this caused little error in the results. The frame coefficient will "normally" be less than 20 percent unless the temperatures on both sides of the specimen are within about 5°C of each other.

The code can be used to calculate the coefficient for any specimen but the more deviation from uniformity the specimen the greater the error will be. Currently, it is necessary to assume or calculate an effective thermal conductivity for the specimen. One area of improvement for the calculations is to incorporate into FLANK a more general model for the specimen.

The code is currently general enough to calculate the various thermal quantities of interest for any homogeneous specimen subjected to constant or time varying temperature boundary conditions. The specimen can be located anywhere on the frame and be of any thickness less than the size of the frame itself.

5. REFERENCES

1. Achenbach, P. R., "Design of a Calibrated Hot-Box for Measuring the Heat, Air, and Moisture Transfer of Composite Building Walls," in Proceedings of the ASHRAE/DOE - OENL Conference on Thermal Performance of the Exterior Envelope of Buildings, December 3-5, 1979, Kissimmee, Florida, p. 308.

2. American Society for Testing and Materials, 1984 Annual Book of ASTM Standards, C976, (American Society for Testing and Materials 1916 Race Street, Philadelphia, PA, 19103, 1984), Part 18.

3. Incropera, F. P. and DeWitt, D. P., "Fundamentals of Heat Transfer," John Wiley and Sons, 1981, pp. 327-329.

4. Lavine, A. G., Rucker, J. L., and Wilkes, K. E., "Flanking Loss Calibration for a Calibrated Hot Box," Research and Development Division, Owens-Corning Fiberglas, Technical Center, Granville, Ohio, 1982, Private Communications.

5. IMSL Subroutine Set, "International Mathematical and Statistical Library." June, 1982, a proprietary package from IMSL, Inc., Houston, Texas.

6. Stewart, G. W., "Introduction to Matrix Computations," Academic Press, New York, 1973, pp 135, 206.

7. Holman, J. P., "Heat Transfer," McGraw-Hill Book Company, 4th Edition, 1976 p. 171.

8. Incropera, F. P. and DeWitt, D. P., "Fundamentals of Heat Transfer," published by John Wiley and Sons, 1981, pp. 140-145.

APPENDIX

USER'S MANUAL FOR COMPUTER PROGRAM - "FLANK"

P. J. Burns
Department of Mechanical Engineering
Colorado State University
Fort Collins, Colorado 80523

R. J. Onega
Department of Mechanical Engineering
Virginia Polytechnic Institute and
State University
Blacksburg, Virginia 24061

Preface

This appendix presents a computer program FLANK which was written in ANSI Standard FORTRAN to calculate the temperature distribution and various heat fluxes through the specimen support frame for the National Bureau of Standards Calibrated Hot Box (CHB). The code is a 2-D transient, finite-difference formulation of the heat conduction equation. The mesh spacing is determined by the user. Either steady-state or dynamic calculations are possible by changing an input parameter into the code. Constant temperature or heat-flux boundary conditions can be imposed on various boundaries of the CHB frame and/or specimen. The transient calculations can be done purely implicitly, explicitly or anywhere between these two methods. Running the code in an implicit mode will eliminate any instability problems.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

1. DESCRIPTION OF THE PROGRAM FLANK

1.1 INTRODUCTION

The program FLANK is a FORTRAN computer program devloped at the National Bureau of Standards (NBS) in Gaithersburg, Maryland, during the summer of 1982. The program was written to calculate temperature and heat transfer quantities associated with the flanking loss for the Calibrated Hot Box (CHB) at the NBS. Researchers at the Owens-Corning Fiberglas Corporation [1] coined the term "flanking loss" for the heat that short circuits around the specimen in the CHB, therefore the name of the program became FLANK.

The program is written to calculate the 2-dimensional conductive heat transfer in the CHB geometry. Thus, the geometry is tied to that of the CHB, but the node structure is general in each of the rectangular elements comprising the CHB geometry. All of the material properties are taken as constant, i.e., independent of time or temperature. Boundary conditions include specification of temperature, or specification of heat transfer coefficients and environmental (ambient) temperatures. The boundary conditions may be constant or functions of time, but not functions of temperature. Either transient or steady-state problems may be simulated.

The program has the capability to calculate a variety of specific heat transfer quantities of interest to the experimentalist such as the heat transferred through the specimen, frame coefficient*, etc. Note that the above assumptions render the problem "linear", although if the heat transfer coefficients are functions of time, it is necessary to reformulate the $\underline{\underline{C}}$ and $\underline{\underline{K}}$ matrices** (to be defined later) at each time step.

This manual is organized to present the theoretical formulation of the problem, to detail the input quantities necessary to run the program and to present example problems. Chapter 1 is an overview of the code. Chapter 2 presents the theoretical formulation of the problem. Chapter 3 details the specific input necessary to run the problem, chapter 4 presents some interactive commands and, finally, chapter 5 presents two example problems.

1.2 SYSTEMS OF UNITS

Where units are specified herein, the SI conventions are used. This is merely for purposes of illustration as any consistent set of units may be used. Also, the user is asked to note that the calculations are done in dimensional form in the program (i.e., no nondimensionalization of the geometry equations is done).

---

* The Frame Coefficient is defined as the fraction of the metering chamber (MC) heat flow that does not go through the specimen.

** Nomenclature: Two bares under a denote that it is a 2-dimensional matrix; one bar denotes a 1-dimensional vector and the remaining quantities are sealars.

1.3 COMPUTER USAGE

The program was written and tested on the Perkin-Elmer 3230 computer of the Center for Applied Mathematics (CAM) at the National Bureau of Standards. The various subroutines were written in ANSI Standard FORTRAN, and thus, should be transportable to other machines. For implementation on another machine, the main program, and the various FORTRAN subroutines must be compiled and loaded together with the IMSL LIbrary (3 IMSL linear equation solving routines are used in FLANK as well as an IMSL plot routine) to generate an executable task controller. Alternatively, the task code already existing on the 3230 may be executed interactively by typing "FLANK" on the user's CRT provided that the card image file "FLANK.INP" (the input file) resides on disk. More details concerning these points are presented below.

1.4 FILES

The following files are necessary to run the computer code on the Perkin-Elmer 3230: (1) FLANK.CSS - the command file, (2) FLANK.TSK - the executable task code, (3) FLANK.INP - the input file, and (4) FTEMP.INP - the restart tempratures*. Upon execution, the following files are generated: (1) FLANK.OUT - the output file, (2) IMSLS.OUT - the IMSL error message file, and (3) FTEMP.OUT - the ending temperatures (see section 3.16). The logical unit association is as follows:

File	Logical Unit No.
IMSLS.OUT	2
FLANK.INP	5
FLANK.OUT	6
FTEMP.INP	1
FTEMP.OUT	3

* The file FTEMP.INP is only necessary to run the code if the restart option is in effect (see section 3.16).

2. THEORETICAL FORMULATION

2.1 THE HEAT DIFFUSION EQUATION

The diffusion of heat in the CHB is completely specified as follows:

$$\rho c_p \frac{\partial^2 T}{\partial t} = k \left[\frac{\partial T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right] \tag{2.1}$$

$$- k \frac{\partial T}{\partial n} = h(T - T_\infty) \text{ on } S_1 \tag{2.2}$$

and

$$T = T(t) \text{ on } S_2 \tag{2.3}$$

$$\text{subject to } T = T_o(y,z) \text{ at } t = 0 \tag{2.4}$$

where y and z denote coordinate directions and t denotes time. Equation (2.1) is the heat balance equation (T is the temperature at position (y, z) at time t) for an isotropic material with constant properties. Equations (2.2) and (2.3) are the convective and temperature specification boundary conditions on surfaces S_1 and S_2, respectively. The entire boundary is composed of the lines S_1 and S_2. Equations (2.1) to (2.3) are the equations that are solved numerically to yield temperatures at the nodal points versus time, viz:

$$\underline{T} = T(y,z,t) \tag{2.5}$$

The equations (2.1) to (2.3) are assembled element by element into a finite difference format and are solved either implicitly ($0 < \alpha \leq 1$) or explicitly if $\alpha = 0$. In matrix notation, the capacitance matrix $\underline{\underline{C}}$ and the conductance matrix $\underline{\underline{K}}$ may be formulated as follows:

$$\underline{\underline{C}} = \rho_i C p_i \, \Delta z_i / \Delta t = C_{ii} \tag{2.6}$$

where the ii denotes that $\underline{\underline{C}}$ is a diagonal matrix, and no sum is intended on the i's, and, denoting either Δy or Δz by ΔX_i, the conductance matrix is

$$\underline{\underline{K}} = k_j \left\{\frac{\Delta X_i}{\Delta X_j} + \frac{\Delta X_j}{\Delta X_i}\right\} = K_{ij}. \tag{2.7}$$

The finite difference (both in space and time) form of equation (2.1) may thus be written for all the nodes as:

$$\underline{\underline{C}}(\underline{T}_{t+\Delta t} - \underline{T}_t) - \underline{\underline{K}}\,\underline{T} = B. \tag{2.8}$$

The right-hand side is the thermal load vector $1\underline{B}$. No mention is made here of the time at which the T for the conductive term is to be taken, as this issue will be addressed in section 2.3. The boundary conditions must also be implemented in the solution scheme. These points will be addressed in the ensuing sections of this manual.

2.2 TEMPERATURE BOUNDARY CONDITIONS

Temperature boundary conditions are handled via a penalty function formulation for generality and ease of implementation. This approach also saves the expense (in coding time) of reformulating the equations by taking the known temperatures to the right-hand side (to the load vector). Thus, if N total node points are specified and M temperature boundary conditions are specified, this approach still requires that N equations always be solved and not N-M. Thus, an increase in formulation and solution time is accepted for ease of implementation.

A brief explanation of the penalty function follows. Suppose that node L is to have its temperature set to T_L^B. In the present scheme, the capacitance matrix has the following term added to the diagonal of row L:

$$\underline{\underline{C}}(L,L) = \underline{\underline{C}}(L,L) + BIG \tag{2.9}$$

where BIG is the penalty function (for the present program BIG = 10^{30}). Since the precision of the present machine is about 10^7, BIG completely overwhelms the original values on that diagonal such that in memory,

$$C(L,L) = BIG \tag{2.10}$$

The same strategy is adopted for the load vector $\underline{B}$, i.e.:

$$B(L) = B(L) + BIG\ T_L^B \tag{2.11}$$

B then is overwhelmed also so that the equation for node L now appears in memory as follows:

$$BIG\ T(L) = BIG\ T_L^B \tag{2.12}$$

which has the solution that $T(L) = T_L^B$ as desired.

The user is simply required to input (1) the vector of node numbers that have temperature boundary conditions and, (2) the vector of the values of the temperatures at the boundary condition nodes. This specific input is described in section 3.10.

2.3 CONVECTION BOUNDARY CONDITIONS

The convection boundary conditions contribute a term to the diagonal of the conductivity matrix, and a term to the load vector. The contribution to the conductivity matrix is:

$$\underline{K}(I,I) = \underline{K}(I,I) + h_I\ \Delta X_I \quad , \tag{2.13}$$

and the contribution to the load vector $\underline{B}$ is

$$B(I) = B(I) + h_I \; T^* \Delta X_i \quad . \tag{2.14}$$

where T* is the ambient temperature and ΔX_I is the length of the surface element exposed to the heat transfer coefficient, h_I. The user is required to input the six values of the heat transfer coefficients and ambient temperatures as defined in section 2.15. The specific input for heat transfer coefficients is described in section 3.9.

2.4 TRANSIENT SOLUTION SCHEME

Letting the solution matrix, C*, be defined as

$$\underline{\underline{C}}^* = \underline{\underline{C}} + \underline{\underline{K}} + BIG_{ii} + (h\Delta X_i)_{ii} \tag{2.15}$$

where the (*) indicates a formuation as appropriate for the solution phase, and equations (2.1) to (2.3) are to be resolved consistent with equations (2.6) to (2.13). This is accomplished as follows.

Define the temperature multiplying the conductivity matrix at the α-point in time, viz:

$$\underline{\underline{K}} \; \underline{T} = \underline{\underline{K}} \left\{ \alpha \; \underline{T}_{t+\Delta t} + (1-\alpha) \; \underline{T}_t \right\} \tag{2.16}$$

Defining:

$$\Delta \underline{T} = \underline{T}_{t+\Delta t} - \underline{T}_t \tag{2.17}$$

The conduction equation becomes

$$\underline{\underline{C}} \; \Delta\underline{T} + \alpha \; \underline{\underline{K}} \; \Delta\underline{T} = - \underline{\underline{K}} \; \underline{T}_t + \underline{B} \; , \tag{2.18}$$

were the term on the right-hand side is known. Incorporating the solution matrix which includes the convective and the temperature boundary conditions leads to the expression

$$\underline{\underline{C}}^* \; \Delta\underline{T} = - \underline{\underline{K}} \; \underline{T}_t + \Sigma \; h_j \; \Delta X_j \; T_j^* + \Sigma \; BIG \; (\Delta T^B)_j \tag{2.19}$$

where the summations indicate assembly into the proper row, j, of the load vector. Note that the penalty function now is multiplied by the incremental temeprature, ΔT_j^B. It is convenient to define the load vector which includes the convection and temperature boundary conditions, and the previous time step conduction contribution, viz:

$$\underline{B}^* = - \underline{\underline{K}T}_t^* + \Sigma_j h_j \; \Delta X_j \; T_j^* + \Sigma \; BIG(\Delta T_j^B) \tag{2.20}$$

Thus Eq (2.19) can be written in the shorthand notation as

$$\underline{\underline{C}}^* \; \Delta\underline{T} = \underline{B}^* \tag{2.21}$$

At the end of each time step, the temperatures are updated via equation (2.17).

There are two solution modes that the transient algorithm can adopt. The first mode to be discussed is the one where the ambient temperatures and temperature boundary condition temperatures are constants, i.e., they are independent of time. In this case, $\underline{B}^*$ is constant with time except for the $K\,\underline{T}_t$ term, and this constant part is formulated (assembled) at the beginning of the solution phase of the program and never reformulated. This assembly is done only once but $K\,\underline{T}_t$ is subtracted at every time step. This solution mode is invoked by settiing ITRAN = 1 on control card 1.

The second mode to be discussed is the one wherein the ambient temperatures and temperature specifications change with time. In this instance, the solution matrix $\underline{B}^*$ is completely reformulated every time step. If this solution mode is desired, it can be invoked by setting ITRAN = -1 on control card 1. During this mode of operation, new values for heat transfer coefficient boundary temperatures and surface temperatures must be provided in the input file for each time step. The input necessary to run a transient problem is presented in section 3.12. No provision has been allowed for the heat transfer coefficients to change with time as this requires reformulation of the C* matrix at each time step. This can consume considerable computer time so this solution technique becomes prohibitively expensive.

2.5 STEADY-STATE SOLUTION SCHEME

Setting ITRAN = 0 on control card 1 will yield a steady-state solution. During this mode of operation, the transient solution scheme is applied for one time step, the previous time step nodal temperatures are all set equal to zero (0), and the integration parameter, α, is set to one (1) so that the incremental temperatures become the steady-state temperatures. The capacitance matrix $\underline{C}$ is effectively removed from the calculation by dividing through by 10^{30}, and the time step is set to one (1) so that the heat transfer output becomes a rate instead of a difference.

2.6 ACCURACY

Accuracy is a function of degree of the time discretization and the spatial discretization of the problem. For the fixed geometry, the spatial discretization is determined solely by the number of node points in the various regions. The user is encouraged to successively increase the number of node points until the quantities in which he is most interested (e.g., heat fluxes or temperatures) change between successive mesh refinements by much less than what the user considers an acceptable amount. This always involves playing off computer time and memory versus accuracy. The user should be aware that smaller mesh spaces result in a quadratic decrease in the stable time step if the time integrated parameter, $\alpha < 1$. In general, the increase in accuracy in a region will vary between linearly and quadratically with the number of mesh points in that region.

Time integration errors are dependent upon both the time step and the time integration parameter. The time step should be reduced just as the mesh must be refined, and the comments in the previous paragraph apply to this point also. The time integration parameter also influences the accuracy. At $\alpha = 0$ and $\alpha = 1$, the time integration scheme is accurate to order Δt. The accuracy gradually shifts from Δt to Δt^2 as α is moved from either $\alpha = 0$ or from $\alpha = 1$ to $\alpha = 1/2$. The scheme for $\alpha = 1/2$ is generally referred to as the Crank-Nicholson scheme [2], and is recommended provided the time step is already small enough to be stable.

2.7 STABILITY

To avoid a geometric increase in the magnitude of the temperatures (and to conform to the second law of thermodynamics), the time step is limited to

$$\Delta t \leq \min \left\{ \frac{\rho C}{(1-\alpha)} \quad \frac{(\Delta X)^2}{k} \right\} \tag{2.22}$$

where the "min" denotes the minimum value overall the grid points and thermal conductivities.

Actually, equation (2.22) only applies at the interior mesh points. At the external mesh points, the convective coefficients exert a debilitating influence on stability. However, the following blanket statement applies: there are no stability problems whatsoever at $\alpha = 1$. So, if stability is critical, the user is encouraged to set $\alpha = 1$ and to accept the slightly lessened time integration accuracy.

2.8 NUMERICAL LIMITATIONS

The calculations are being performed on a 32-bit machine in the single precision mode. The structure of a floating point number limits the accuracy to about 7 decimal places and the limit of the magnitude is about 10^{75}. In some instances, 7 places of accuracy will be lost during the IMSL solution of the linear set of equations even though full pivoting is allowed. If this occurs, the IMSL routine will return a nonzero value of the IER parameter, and the program will automatically terminate after writing a brief explanation of the error condition into the output file FLANK.OUT. A more complete description of the problem can usually be found in the file IMSLS.OUT wherein the IMSL routine writes its own internally defined comments.

If this situation occurs, the user has two options: (1) decrease the number of node points and rerun the problem or (2) reload the program in double precision mode.

During the course of decomposing the solution matrix, the IMSL routine LUDATF computes the determinant to test for singularity (when the determinant = 0). It is very likely that the determinant will exceed 10^{75} in value and generate an overflow of the exponent register (this does not affect the solution as this value is never used). When this happens, an interactive message is sent to the user at his CRT. It is likely that this is the place where the overflow occurs, but it might not be the only place. Consequently, the user is encouraged to check his answers very carefully whenever this happens.

It is also likely that an underflow will be generated during the decomposition and/or refinement phase of the IMSL solution. A message to this effect is displayed upon the user's CRT and may be ignored.

2.9 ORGANIZATION OF THE PROGRAM

The program is logically organized into 3 basic elements. These elements are capable of being overlayed, but were not as very little can be gained from this. The 3 elements are as follows:

(1) Main routine
(2) Input routine
(3) Solution routine

The main routine sets the storage for the common arrays, vectors and constants, and then controls the program flow by calling the input and solution routines sequentially.

The input routines consist of a variety of subroutines to read and echo all of the input parameters. The reader may consult the code listing for more specific information.

The solution routines perform a variety of functions, including the following basic ones:

(1) assembly of the solution matrix, $\underline{C}^*$
(2) assembly of the load vector, $\underline{B}^*$
(3) performance of the energy balances and tallies
(4) updating and printing the solution vector, $\underline{T}(t+\Delta t)$

and

(5) solving the linear algebraic equations including:

(a) decomposition of the solution matrix, $\underline{C}^*$
(b) back substitution to obtain $\underline{T}(t+\Delta t)$

and
(c) refinemenet to obtain a more accurate $\underline{T}(t+\Delta t)$.

For the finer details, the user is referred to the code listing.

2.10 RESTART CAPABILITIES

Occasionally, the user may wish to run the program in the transient mode for a short period of time, stop, examine the answers and start the program from where the last calculation stopped (restart). In order to facilitate this process, FLANK has the option to write a file of nodal temperatures upon termination (file name = FTEMP.OUT). This file may later be renamed and read as the initial temperatures for the subsequent run (file name = FTEMP.INP). This capability eliminates the laborious manual editing necessary to feed these temperatures into the input file so that the user need only change the starting and ending times (and some control parameters) in the input file. The specific input required to exercise this option is detailed in sections 3.1 and 3.12.

2.11 DATA CHECK ONLY

One of the common errors encountered when using the computer occurs when the data are placed incorrectly in the input file. This is particularly critical when the input data control the memory location of subsequent input datums (e.g., supposing that a temperature boundary condition is to be supplied to node 5 in an I5 format, suppose further that the input is not right-justified and that 50000 is read in lieu of 5, the value of the boundary temperature read subsequently will then be stored in the location of the start of its array + 50000 and this will certainly exceed the length of the array with the result that something, either a storage value or even executable code, will get over-written). To aid the user in debugging such errors, FLANK has the capability to read in and echo the input data, and to then stop so that the user may print out the output file and check it for such errors. This capability is invoked by setting IDATA = 1 on control card 1 as is described in section 3.2.

2.12 COORDINATE SYSTEM AND GEOMETRY

The coordinate system for the problem is shown in figure 2.1 with X_1 = y and X_2 = z. The origin exists as the bottom left-hand corner of the base of the CHB. The geometry is 2-dimensional with a horizontal y-axis and a vertical z-axis. All of the dimensions shown in the figure are adjustable and must be input by the user as described in sections 3.6 and 3.8.

2.13 NODAL DATA

The associated nodes are shown in figure 2.2. Note that the total number of nodes must be specified in the first horizontal and the first vertical sections (NZPOLY and NY5, respectively) so that nodes are specified at the starting boundary line and the ending boundary line in these sections. In all other sections, one line of boundary nodes already exists, from the specification of the previous section. Stated another way, in the first horizontal section and first vertical section:

Entering N nodes = N-1 grid spaces

In all other sections

Entering N nodes = N grid spaces.

Input of this type is detailed in sections 3.5 and 3.7.

2.14 MATERIAL REGIONS

The material regions are also shown in figure 2.1. Normally, the following materials will be specified by number:

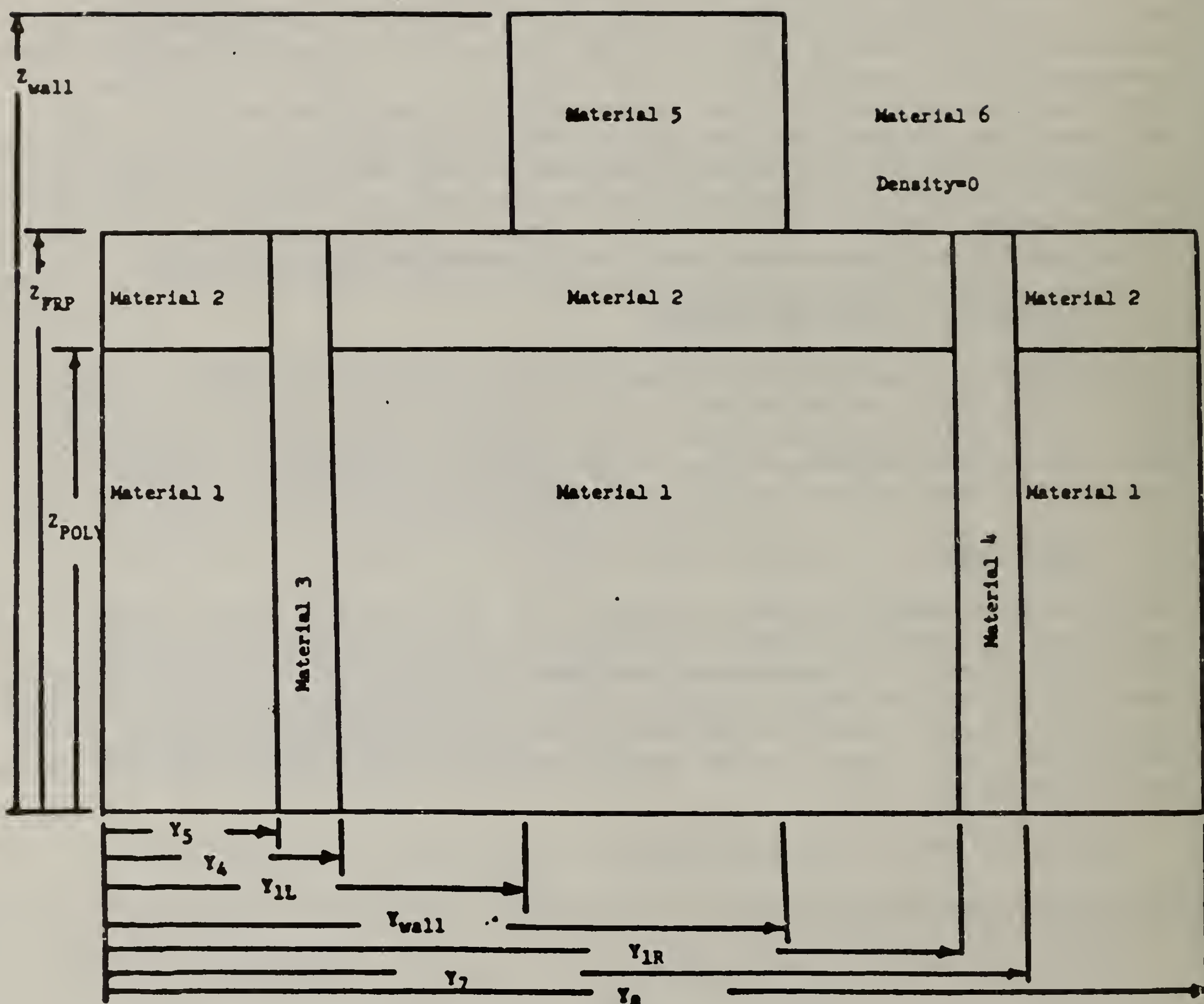

Figure 2.1 Coordinate system showing the input dimensions required for FLANK and the various material regions

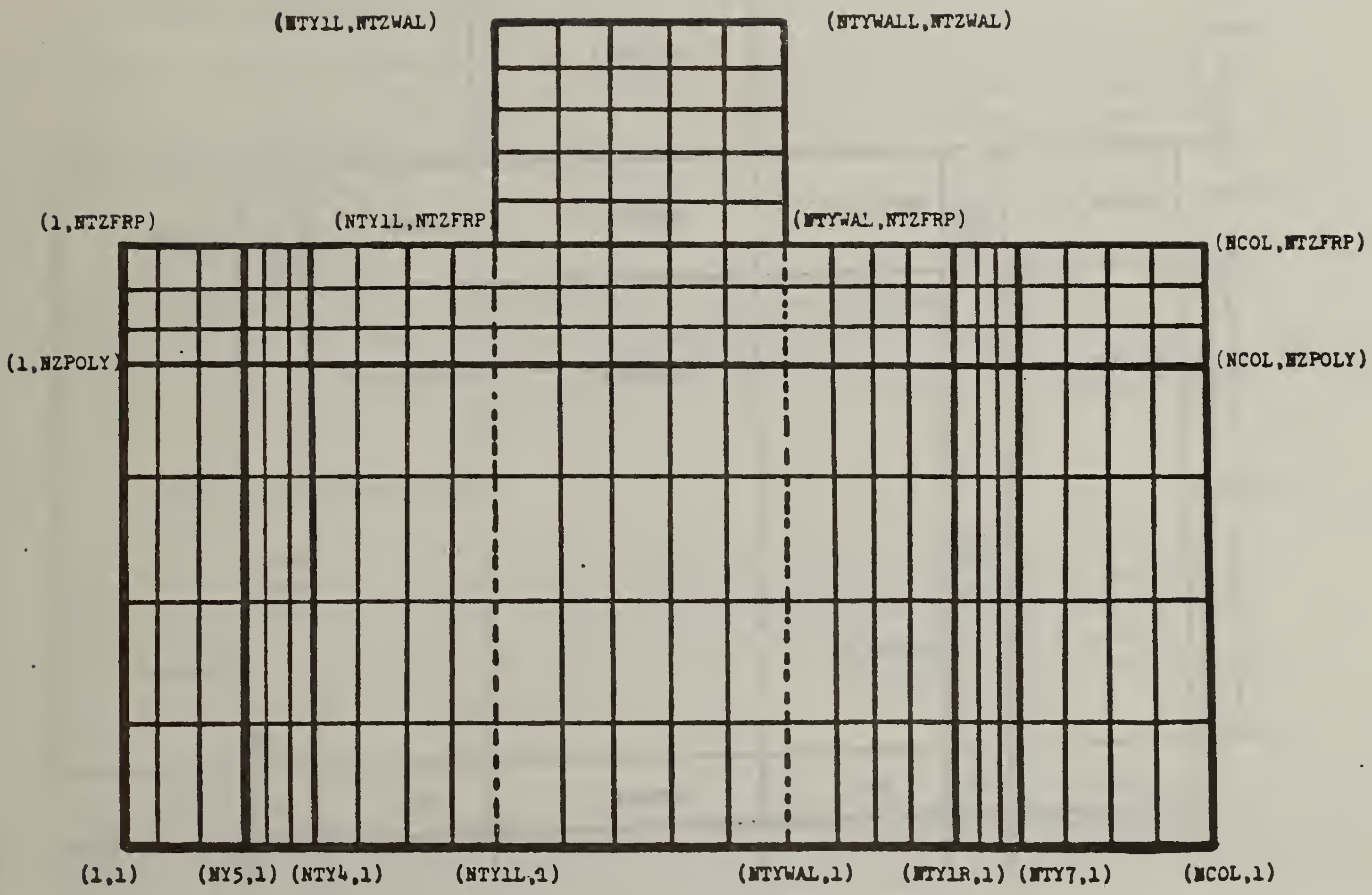

Figure 2.2 The node structure for the NBS calibrated hot box

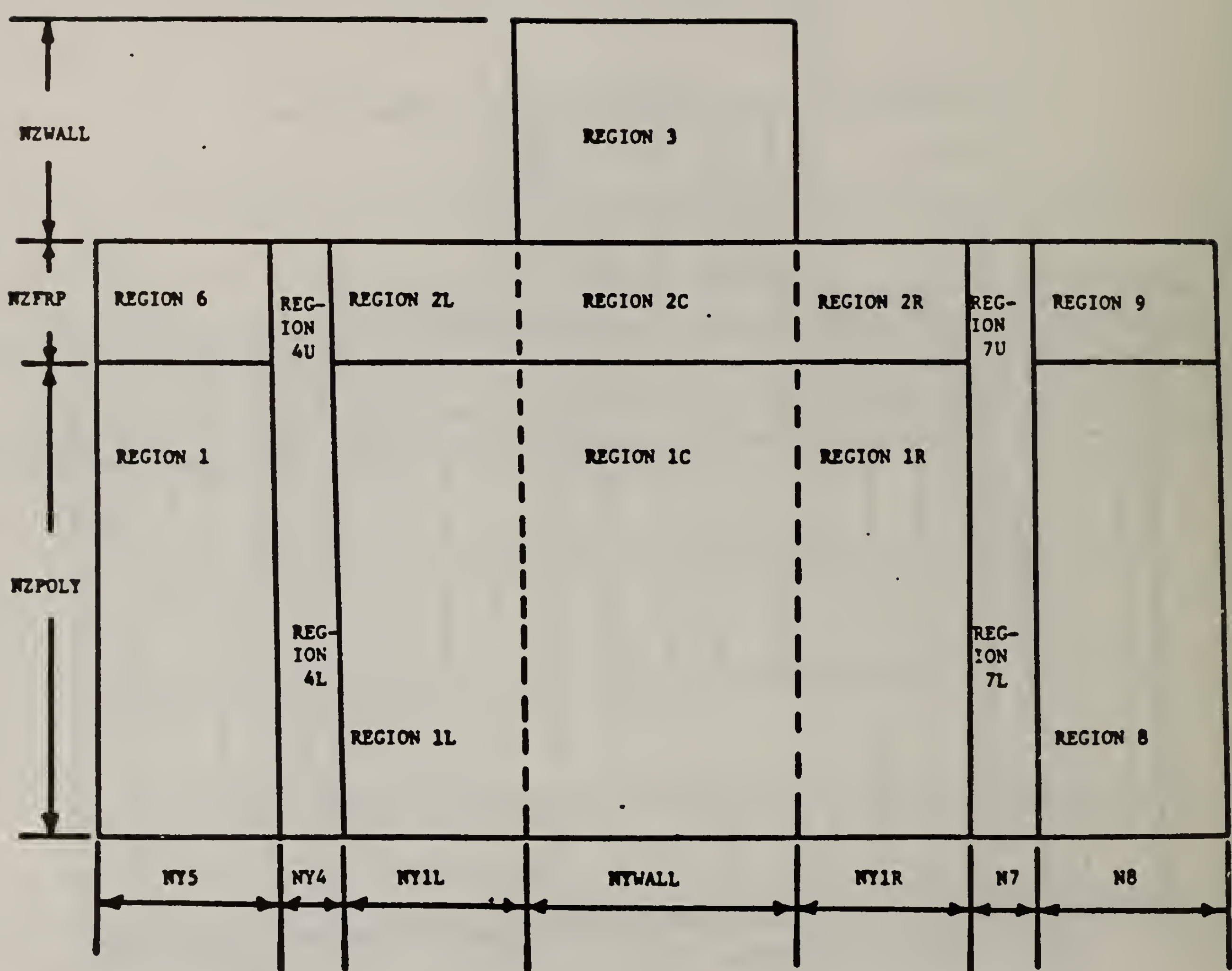

Figure 2.3 The various node designations for each of the geometrical regions of the wall specimen and frame

Materials No.	Type
1	Polyurethane
2	Fiberglass Reinforced Polyester (FRP)
3	Fixed Gasket (Neoprene)
4	Inflatable Gasket
5	Wall
6	Null

It is required to input values for the thermal conductivity, the density and the specific heat. Only the first five (5) materials may be input; the sixth is internally set. This is detailed in section 3.10.

2.15 HEAT TRANSFER COEFFICIENT REGIONS

The heat transfer coefficients are shown in figure 2.4 by region. Note that a counterclockwise sequencing has been adopted. Normally, the heat transfer coefficents for regions 2, 5, and 8 will be set to zero to represent an insulated (zero heat flow) condition. The heat transfer coefficient input data are described in section 3.9.

2.16 HEAT FLOW LINES

There are 13 lines along which heat flows have been defined. These lines are shown in figure 2.5 where the heat flow has been defined as positive in the positive direction of the coordinate axis (except for convective heat transfer). The input to invoke the energy balance routines is described in section 3.2 and 3.4.

2.17 ENERGY BALANCE REGIONS

There are 5 regions wherein steady-state energy balances may be performed. These are also shown in figure 2.5 as enumerated areas. The user is cautioned that specifying a temperature boundary condition on or in a region adds or subtracts energy, and that this is not explicitly accounted for in the energy balance routine resulting in a large net flow for that region (i.e., the energies seem not to balance for this reason).

2.18 MATERIAL SUBREGIONS

The subregions are the result of the grid application in that everywhere an area is bounded by coordinate lines, the grid inside the coordinate lines is regular. These regions are smaller than the material regions detailed in section 2.14, and, hence, have been termed subregions. The B, U, L, R and C specification has been adopted for bottom, upper, left, right, and center wherever possible. These subregions will ordinarily be transparent to the user, but have been included here for completeness.

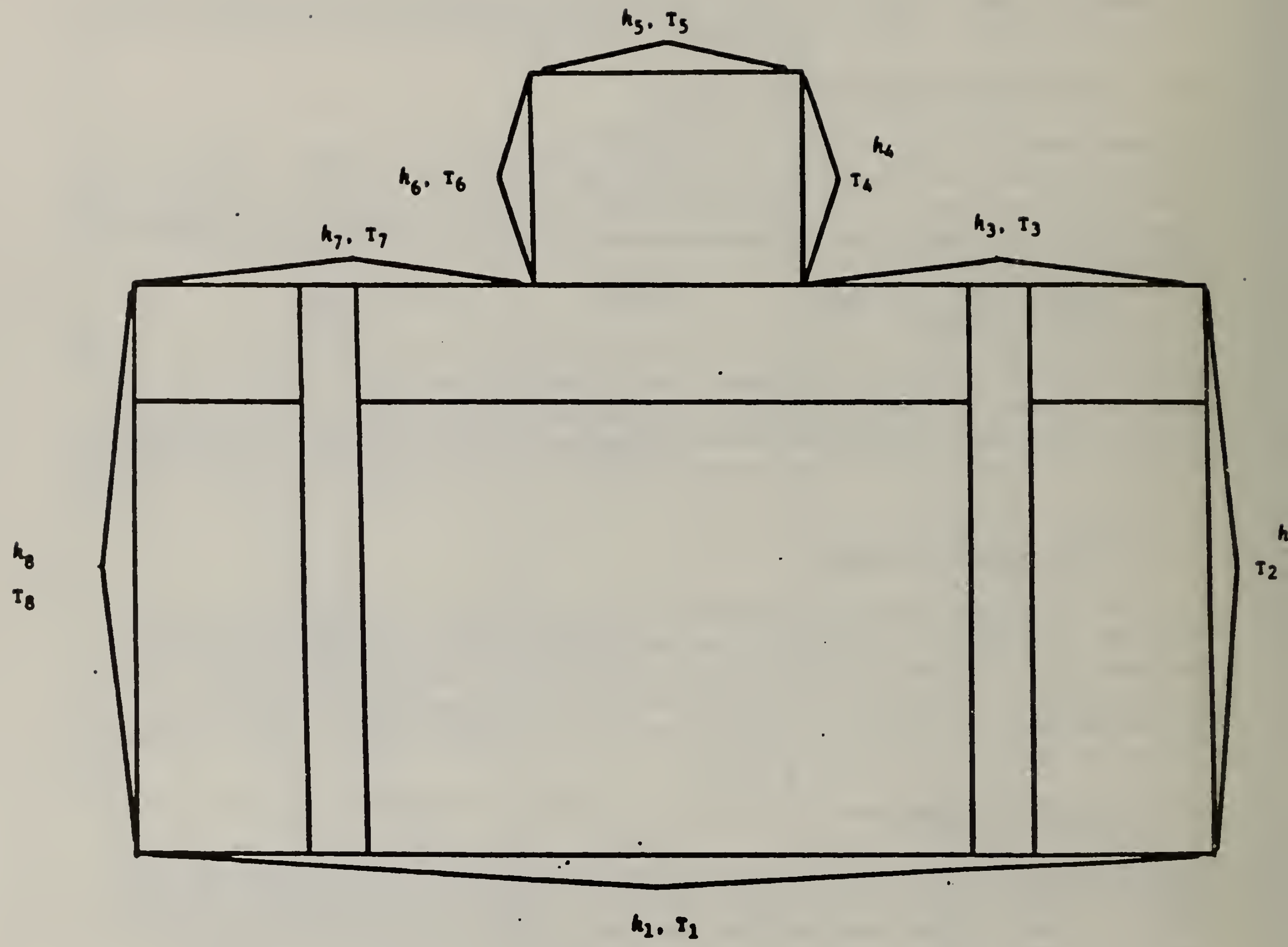

Figure 2.4 Heat transfer coefficients and associated temperature designations

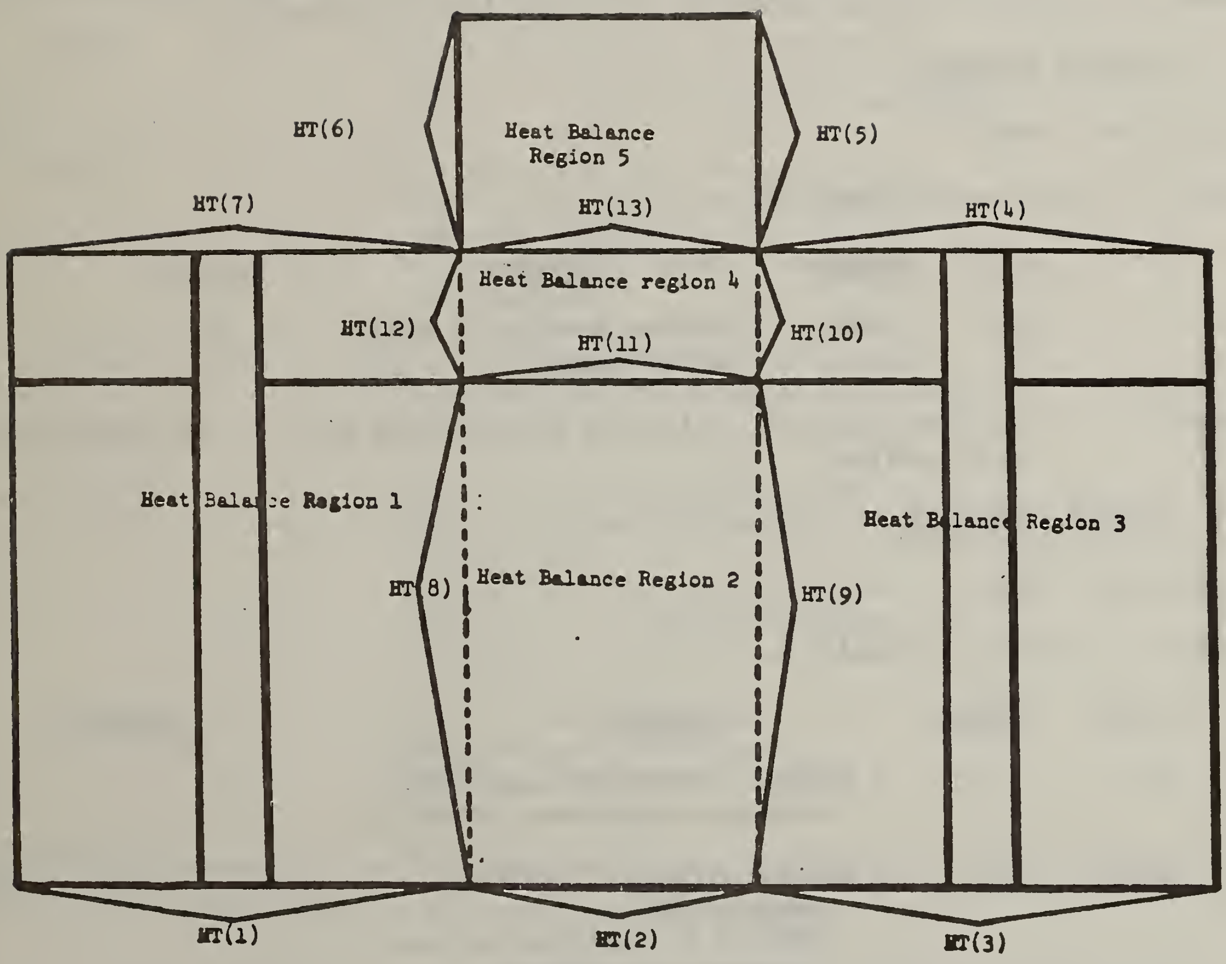

Figure 2.5 The heat flow lines and heat balance region designations

3. INPUT DATA STRUCTURE

The following sections detail the specific input required to run the program. The input is organized into the following sequential areas:

(1) Problem Heading - 1 Card
(2) Control Information - 1 Card
(3) Constant Data - {(12 + NUMTBC) cards, steady-state; (13 + NUMTBC) cards + initial temperature data (variable number of cards), transient
(4) Transient input required if the heat transfer coefficients and ambient temperatures change with time -2 (number of time steps) cards.

The control information is so termed because these data control both the number of data items read and the execution of the program. The input follows in the sequential fashion that is necessary for input into the program.

3.1 PROBLEM HEADING

Conditions - None

Number of Cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-64	16A4	Problem heading to appear on each page	1

Note(s): 1. The user may wish to include the time and date in the heading for each problem.

3.2 CONTROL DATA CARD

Conditions - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-5	I5	NUMTBC - number of temperature boundary conditions, NUMTBC $\geq$ 0.	
6-10	I5	NUMIT - number of initial temperatures NUMIT < 0 - uniform initial temperature = TEMPØ NUMIT $\geq$ 2 - NUMNP initial temperature read from file FTEMP.INP	1,2

Cols.	Format	Item(s)	Note(s)
11-15	I5	IDATA - data check only code IDATA = 0 - normal execution IDATA ≠ 0 - data check only	
16-20	I5	IPR - print interval, temperature output is printed every IPR time steps	
21-25	I5	IRFINE - refinement control parameter IRFINE = 0 - no refinement is done IRFINE ≠ 0 - refinement is done	3
26-30	I5	IDGT - IMSL accuracy test parameter IDGT = 0 - no accuracy test is done IDGT > 0 - accuracy test is performed	4
31-35	I5	IEB - heat flow parameter IEB = 0 heat flows or energy balances are not computed IEB ≠ 0 energy balances and heat flows are computed	
36-40	I5	IQPR - energy tally print interval, energy tallies are printed every IQPR time steps.	
41-45	I5	ITRAN - transient solution code IRAN = 1 - transient problem, constant parameters ITRAN = 0 steady state problem ITRAN = - 1 - tansient problem, variable parameters	5
46-50	I5	IPLT - plotting parameter IPLT = 0 no plots done IPLT = 1 plots done	

Note(s):

1. If ITRAN = 0, no initial temperatures are read as the initial temperatures are automatically set equal to 0.

2. If NUMIT $\leq$ 2, the file FTEMP.OUT which includes the nodal point temperatures is written. The name of this file can be changed to FTEMP.INP (provided any existing FTEMP.INP is first deleted) which can be read on a subsequent problem if NUMIT $\geq$ 2.

3. During refinement, the output vector obtained from back substitution is iterated upon with the original solution matrix (the decomposed

solution matrix is used to obtain the output vector by back substitution) to achieve greater accuracy. This endeavor is usually worthwhile as the cost is not great.

4. Upon decomposition, a check is made to determine if IDGT's of accuracy were retained in the decomposed mode. If IDGT's of accuracy were not retained, an error termination results. This is not a crucial test, as refinement tends to improve the answers apart from the decomposition mode. Therefore, IDGT = 0 is recommended.

5. For ITRAN = -1, new values are required for all temperatures at the temperature boundary condition nodes and for all ambient temperatures in the enclosure. The capability does not presently exist to have variable heat transfer coefficients although this can be handled as a restart option (it is rather awkward, though).

3.3 DEBUG INFORMATION

Conditions - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-50	10I5	IDBUG(10) - debug control array. If IDBUG(I) is different from zero, the following information is printed. I = 1 $\underline{K}$ and $\underline{C}$* matrices, $\underline{B}$* vector I = 2 - decomposed C* matrix I = 3 - nodal association data I = 4 - element assembly matrices I = 5 - interior data written I = 6 - $\underline{C}$* matrix written from equation solver subroutine I = 7 - $\underline{B}$* vector written from equation solver subroutine I = 8 - I = 9 - I = 10 -	1

Note(s): 1. Normally, all values should be set to zero (Ø), as this information was required during the debug phase which is presently complete.

3.4 HEAT BALANCE CHECKS

Conditions - IEB ≠ 0

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-25	5I5	ICHECK(5) - energy balance control array. If ICHECK(I) $\neq$ 0, the following steady-state energy balance checks are performed and printed, (see figure 2.5 for these heat balance regions). I = 1 - region 5-6-4U-4L-1L-2L I = 2 - region IC I = 3 - region 1R-2R-7L-7U-8-9 I = 4 - region 2 C I = 5 - region 3	

3.5 HORIZONTAL NODE NUMBER SPECIFICATION

Conditions - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-10	E10.0	NY5	1,2
11-20	E10.0	NY4	3
21-30	E10.0	NY1L	3
31-40	E10.0	NYWALL	3
41-50	E10.0	NY1R	3
51-60	E10.0	NY7	3
61-70	E10.0	NY8	3

Note(s): 1. See section 2.13 for definitions of the nodal specifications.

2. This number represents the (number of grid spaces - 1) in the region.

3. This number represents the number of grid spaces in the region.

3.6 HORIZONTAL COORDINATE SPECIFICATION

Conditions - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-10	E10.0	Y5(m)	1
11-20	E10.0	Y4(m)	
21-30	E10.0	Y1L(m)	
31-40	E10.0	YWALL(m)	
41-50	E10.0	Y1R(m)	

Cols.	Format	Item(s)	Note(s)
(cont.)			
51-60	E10.0	Y7(m)	
61-70	E10.0	Y8(m)	

Note(s): 1. The reader is referred to section 2.12 for definitions of these coordinates (see figure 2.1).

3.7 HORIZONTAL NODE NUMBER SPECIFICATION

Condition - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-10	E10.0	NZPOLY	1,2
11-20	E10.0	NZFRP	3
21-30	EL0.0	NZWALL	3

Note(s): 1. The reader is referred to section 2.12 for definitions of the node numbers.

2. This number represents the (number of grid spaces -1) in the region.

3. This number represents the number of grid spaces in the region.

3.8 VERTICAL COORDINATE SYSTEM

Conditions - None

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-10	E10.0	ZPOLY(m)	1
11-20	E10.0	ZFRP(m)	
21-30	E10.0	ZWALL(m)	

Note(s): 1. The reader is referred to section 2.12 for definitions of the coordinates (see figure 2.1).

3.9 HEAT TRANSFER COEFFICIENT INPUT

Conditions - none

Number of cards to be read - 2.

Cols.	Format	Item(s)	Note(s)
		(First Card)	
1-80	8E10.0	HTC(I), heat transfer coefficients, input sequentially by regions, I($W \cdot m^{-2} \cdot K^{-1}$).	1
		(Second Card)	
1-80	8E10.0	TAMB(I), ambient temperatures, input sequentially by region, I (K).	1

Note(s): 1. The reader is referred to section 2.14 for definitions of the heat transfer coefficient regions (see figure 2.4).

3.10 MATERIAL INPUT

Conditions - none

Number of cards to be read - 3.

Cols.	Format	Item(s)	Note(s)
		(First Card)	
1-50	5E10.0	RKAY(I), thermal conductivity, input by material region, I ($W \cdot m^{-1} \cdot K^{-1}$)	1
		(Second Card)	
I-50	5E10.0	RHO(I), densities, input by material region, I ($kg \cdot m^{-3}$)	1
		(Third Card)	
1-50	5E10.0	CP(I), specific heats, input by material region, I ($J \cdot kg^{-1} \cdot K^{-1}$)	1,2

Note(s): 1. The reader is referred to section 2.14 for definitions of material regions.

2. Note that the units must be input as Joules and not kilojoules.

3.11 TEMPERATURES BOUNDARY CONDITION INPUT

Conditions - NUMTBC ≠ 0

Numbers of cards to be read - NUMTBC/5.

Cols.	Format	Item(s)	Note(s)
1-75	5(I5, E10.0)	(NODEST(I), TBC(I), I=1, NUMTBC) (the node number at which the temperature is applied, and the applied temperature (K))	1,2,3

Note(s): 1. Temperatures may be applied at any nodes whether they are on the boundary or not.

2. Node numbers vary sequentially along rows (by column) starting with 1 at the origin. The node numbers are incremented by one in all cases. The sequence varies by row through the entire base. The next sequence starts at the lower left-hand of the wall specimen and is incremented by one along the rows (again by column) through the wall.

3. An example of nodal numbering is given in chapter 4.

3.12 TIMING INPUT

Conditions - ITRAN ≠ 0

Number of cards to be read - 1.

Cols.	Format	Item(s)	Note(s)
1-10	E10.0	TIMØ - initial time(s)	
11-20	E10.0	TIMF - final time(s)	
21-30	E10.0	DELTA - time step(s)	1
31-40	E10.0	ALFA - time integration parameter, Ø < ALFA ≤ 1	2

Note(s): 1. The number of time steps to be taken will be = (TIMF - TIMØ)/DELTA.

2. The reader is referred to sections 2.6 and 2.7 for recommended values of ALFA.

3.13 INITIAL TEMPRATURE INPUT

Conditions - NUMIT ≠ 0

Number of cards to be read - 1 or NUMNP/5.

Cols.	Format	Item(s)	Note(s)
		(Case 1, NUMIT = -1)	
1-10	E10.0	TEMPØ - uniform initial temperature (K), (1 card only read)	1,2

Cols.	Format	Item(s)	Note(s)
(cont.)			
		(Case 2, NUMIT = 1)	
1-75	5(I5, E10.0)	(NODE, T(NODE), I = 1, NUMNP), node number and initial temperature of that node (K), (NUMNP/5 cards read)	2

Note(s): 1. Every node point temperature is initially set ot TEMPØ.

2. For reading initial temperatures from the disk file FTEMP.INP, NUMIT should be set to 2. For writing final temperatures into the disk file FTEMP.OUT, NUMIT should be set to -2. This file may be renamed to FTEMP.INP and subsequently read. (See sections 2.10 and 3.2.)

3.14 BASE PLOT DATA

Conditions - IPLT = 1

Numbers of cards to be read - 2.

Cols.	Format	Item(s)	Note(s)
		Card 1 - Plot Control	
1-5	I5	NUMCON - number of contour plots to be plotted $0 \leq$ NUMCON ≤ 5	1
6-10	I5	NUMHIS - number of time history plots to be plotted $0 \leq$ NUMHIS ≤ 5	1
11-15	I5	IOPT - plot column option IOPT = 0 - narrow plots (80 columns) IOPT = 1 - wide plots (126 columns)	2
16-20	I5	NOUT - plot output file specifier NOUT = 2 - plots written to file IMSLS.OUT NOUT = 5 - plots written to output file FLANK.OUT	3
21-25	I5	NSTEPC - time step interval for contour plots (5 time curves are accumulated before the plot is written, this is 5 * NSTEPC time steps)	4

Note(s): 1. A maximum of 5 curves is permitted.

2. The IOPT = 0 plots are especially convenient for tracing on 8 1/2" x 11" paper.

3. Reading an NOUT different from 2 or 5 will result in an error termination.

4. The contour plots are written every 5 time stpes or at the end of the problem, whichever comes first.

Cols.	Format	Item(s)	Note(s)
		Card 2 - Range Data	
1-10	E10.0	TMIN - the minimum temperature value for the vertical plot axis	1,2
11-20	E10.0	TMAX - the maximum temperature value for the vertical plot axis	2,3

Note(s): 1. All plotted values should be above TMIN.

2. It is a good idea to allow some leeway in picking the range for the plots as it is convenient to have some space between the plotted values and the axes.

3. All plotted values should be less than TMAX.

3.15 CONTOUR PLOT DATA

Conditions - NUMCON $\neq$ 0

Number of cards to be read - 5 x (NUMCON).

Contour plots are "snapshots" of the temperature profile along specified node points at an instant in time. Five curves of temperature versus node point number are plotted at 5 specified times (unless the last time step is taken, in which case the plots are done before termination), on any one curve. A maximum of 5 plots are allowed.

The following block of input items is required for each plot to be drawn. The blocks should be input sequentially, until NUMCON of them have been input.

Cols.	Format	Card No.	Item(s)	Note(s)
1-72	72A1	1	ITITC(I) - plot title	
1-36	36A1	2	ITITXC(I) - x-axis title	
1-36	36A1	3	ITITYC(I) - y-axis title	
1-5	I5	4	NC(I) - number of nodes against which temperatures are to be plotted, $2 \leq NC(I) < 20$	1
1-80	16I5	5	NODESC(I), the node numbers of the temperatures to be plotted	2,3

Note(s): 1. A maximum of 20 nodes is permitted.

2. Actually, 2 cards may be required for this input in which case 6 x NUMCON cards must be input.

3. The nodes may be input in any order, i.e., the nodes need not be adjacent.

3.16 TIME HISTORY PLOT INPUT DATA

Conditions - NUMHIS $\neq$ 0

Number of cards to be read - 5 x (NUMHIS)

Time history plots are plots of the node temperature versus time. The abcissa is fixed as a linear variation of time from the initial time to the final time. Up to 5 node numbers may be included on each plot. A maximum of 5 plots is allowed. Unlike the contour plots, only 5 plots are written to the plot file (the contour plots allow up to 5 plots to be written every 5 x (NSTEPC) time steps).

The following block of input items is required for each plot to be drawn. The blocks should be input sequentially until NUMHIS of them have been input.

Cols.	Format	Card No.	Item(s)	Note(s)
1-72	72A1	1	ITITH(I) - plot title	
1-36	36A1	2	ITITXH(I) - x-axis title	
1-36	36A1	3	ITITYH(I) - y-axis title	
1-5	I5	4	MH(I) - number of node points to be included on this plot $1 \leq MH(I) \leq 5$	1
1-80	16I5	5	NODESH(I) - the node numbers of the temperature to be plotted	2,3,4,5

te(s): 1. A maximum of 5 nodes is allowed.

2. The nodes may be input in any order.

3. Only 1 plot for each block of nodes is allowed for the entire time interval.

4. The number of stored values is internaly limited to 80, maximum. If the total number of time steps divided by 80 is not an integer number, less data are stored (e.g., for 100 time steps, 50 data points are stored).

5. The starting time temperatures and the ending time temperatures are automatically included in the plots.

3.17 TRANSIENT DATA INPUT

Conditions - ITRAN = - 1

2 sets of data to be read for each time step.

Cols.	Format	Item(s)	Note(s)
		First Data Set - NUMTBC/8 cards to be read	
1-80	8E10.0	(TBC(I), I=1, NUMTBC) - temperature boundary condition temperatures, input sequentially (K).	1,2
		Second Data Set - 1 card	
1-80	8E10.0	(TAMBP(I), I=1, 8) - ambient temperatures at the end of the time step (K).	2,3

Note(s): 1. The nodes at which the boundary temperatures were set (NODEST(I)) remain unchanged.

2. Complete sets of these data must be input at each time step, regardless of whether or not the values change.

3. The heat transfer coefficients (HTC(I)) remain unchanged (at their initially input values).

4. RUNNING THE PROGRAM

4.1 INTERACTIVE EXECUTION

A Command Substitution System (CSS) file, FLANK.CSS, is available to run the program interactively on the Perkin-Elmer 3230. The file loads the executable task code, FLANK.TSK, into memory, allocates disk space for output files, assigns local unit numbers, and prints the timing log at the user's terminal screen at the end of execution. Interactive execution is accomplished by typing

FLANK

followed by a (carriage) return.

The following files must be available for the program to read from:

(1) FLANK.INP - the input file as described in chapter 3.

(2) FTEMP.INP - the file of initial temperatures in a 5E14.Ø format (this file may contain only 1 blank line if no reads are required, i.e., NUMIT $\neq$ 2.

The following output files are generated:

(1) FLANK.OUT - the output file with the echo of the input data, the temperature states, the energy balance information, and the debug information.

(2) FTEMP.OUT - the output file of the final temperatures written in a 5E14.6 format.

(3) IMSLS.OUT - the output file containing any IMSL debug information.

Should difficulty be encountered in running the computer code, the user should locate and inspect the files FLANK.CSS, FLANK.INP, FTEMP.INP, and FLANK.TSK. Usually, the problem will be that one of these files is missing, or the data are input incorrectly in FLANK.INP. If problems are still encountered, the user should try a data check only (see section 3.2).

4.2 ERROR TERMINATION

The program internally checks for a variety of input errors. If any input errors are detected that are of such a nature as to prohibit the execution of the program, an error termination is invoked. Then, a descriptive message is printed in the file FLANK.OUT, together with a line denoting:

" - - - ERROR TERMINATION - - - "

One of the most common mistakes is easy to decipher. This mistake occurs when the user has input more node points than the allowed storage locations. The following message is then printed:

```
"NO. OF NODE POINTS = " N1
"EXCEEDS DIMED VALUE = " N2
"STOP"
```

where N1 is the number of node points that the user has specified, N2 is the maximum number allowable, and DIMED is an abbreviation for "dimensioned". Should this occur, the user has only 2 choices: (1) reduce N1 to be $\leq$ N2, or (2) edit the FORTRAN source code FLANK.FTN to increase N2 (NNODES in FLANK.FTN) and reload the program using the LINK33.CSS routine on disk.

4.3 USEFUL UTILITY ROUTINES

Some useful system utility routines that exist on the NBS Perkin-Elmer 3230 computer will be described in this section. All lines must be input exactly as shown, followed by a carriage return. All lines contain parameters that will also be described. Optional parameters will be included in parenthesis. The parameter PR: is the system definition for the line printer, and when included, will automatically send the output to the line printer. The utility routines are enumerated, and the purpose of each routine is given.

4.3.1 The Rename Routine

Command Line: REN fn1.ex1,fn2.ex2

Parameters: fn1.ex1 - file name fn1 with extension . ex1

fn2.ex2 - file name fn2 with extension. ex2 (fn2.ex2 must not exist on disk).

Purpose: REN renames a file. This may be useful when a file is to be printed (e.g., FLANK.OUT) for when a file is in the print queue, it cannot be deleted or rewritten. Also, the file of output temperatures may be renamed to be the input temperatures for a subsequent run.

Examples: REN FTEMP.OUT, FTEMP.INP

REN FLANK.OUT, F1.OUT

4.3.2 The Delete Routine

Command Line: DE fn1.ex1

Parameters: fn1.ex1 - the file fn1.ex1 is deleted from disk

Examples: DE FTEMP.INP

4.3.3 The Print Routine

Command Line: PRINTfnl.exl

Usage: PRINTFl.OUT

4.3.4 The Compile Routine FORT7C

Command Line: FORT7C fnl, (PR:), (XREF), 10, (lib.exl)

Parameters: fnl - the file of FORTRAN source code fnl.FTN (note the extension is not given on the command line, and it must be FTN) is compiled, file fnl.OBJ (the object code) is generated and written to disk.

PR: - the compiler listing is sent to the printer

XREF - the cross reference map is sent to the printer

lib.exl - the library file lib with extension, .exl=.LIB, is loaded also. (For compiling FLANK.FTN, this space should be left blank.)

Examples: FORT7C FLANK, , , 10

FORT7C FLANK, , XREF, 10,

4.3.5 The Compile/Library Load Routine FORT7CR

Command Line: FORT7CR fnl, (PR:), 10

Parameters: fnl - the FORTRAN source code file fnl.FTN is complieted

PR: - the compiler listing is sent to the printer.

Purpose: The object code (compiler output)

fnl.OBJ is generated from the compiler. If this routine exists in the library FLANK.LIB, it is deleted. The routine fnl.OBJ is then written into the object code library FLANK.LIB. This object code library is loaded together with FLANK.OBJ, the IMSLS library, and the system FORTRAN library by the utility routine LINK33.CSS.

Example: FORT7CR EQSOL2, , 10

FORT7CR EQSOL2, PR:,10

4.3.6 The Linker LINK33.CSS

Command Line: LINK33 FLANK, (PR:), FLANK.LIB

Parameters: FLANK - the main routine object code file FLANK.OBJ is loaded

PR: - a cross reference map of the load is sent to the printer

FLANK.LIB - the object code files of all of the subroutines are loaded from the object code library FLANK.LIB.

Purpose: This routine loads all of the object code, together with the IMSLS library and the system FORTRAN library to generate the executable task code FLANK.CSS.

Usage: LINK33 FLANK, , FLANK.LIB

LINK33 FLANK,PR:, FLANK.LIB

4.3.7 The DELELIB Utility Routine

Command Line: DELELIB fn1, fn2.LIB

Parameters: fn1 - the object code file fn1.OBJ is deleted from the object library fn2.LIB

fn2 - the object code library fn2.LIB

Usage: DELELIB FLANK, FLANK.LIB - this removes the routine FLANK.OBJ from FLANK.LIB. Note that this file should <u>not</u> be in the library as it gets loaded together <u>with</u> the library at link time via LINK33.

4.4 <u>CRT MESSAGES</u>

When executing the program, various error messages and other messages will appear on the CRT terminal. If an error does occur, the user must then cancel the job using the command

```
CA
```

The following examples are listed, together with an explanation.

4.4.1 Input Data Incorrect

If the input data are not in the correct format or if data are missing, the following sequence is likely to be seen on the screen.

```
* FLANK - program initiation step
- IN SOLVE2 - AT BEGINNING
- IN SOLVE2 - CALLING ASEMB2
- IN SOLVE2 - ASEMB2 DONE, CALLING FORMB
- NON-EXISTENT SEGMENT ERROR (PST) AT B4C80
- MEMORY FAULT ADDRESS = FFFD30
TASK PAUSED
*
```

This error message resulted because the number of temperature boundary conditions NUMTBC was 3 but only two temperatures were actually supplied to the input file. The only corrective action necessary is to edit FLANK.INP and change NUMTBC to 2 or else supply a third node specification and temperature for a boundary point.

4.4.2 Underflow Errors

Often with specified temperature boundary conditions, the message

```
- FLOATING POINT - UNDERFLOW ERROR AT D70D2
- NEXT INSTRUCTION AT D70D8
```

will occur. The D70D2 are the hexadecimal memory locations for the particular subroutine in which the underflow occurred. These errors can be ignored on the P-E since penalty functions are being used and division by a very large number occurs. The PE simply sets the result to zero.

4.4.3 Overflow Error

The determinant of the $\underline{\underline{C}}$ or $\underline{\underline{K}}$ matrix will often be greater than 10^{76} if a large number of nodes is used. The typical value of a diagonal element is about 2 so that if there are 246 nodes, $2^{246} = 10^{74}$. This determinant value is computed but never used so the error may be ignored. The CRT screen will display the following message

```
ERR 517 (OC 94A6):
RXXR: EXPONENTIAL OVERFLOW
IN SOLVE 2 - DONE DECOMPOSITION, BEGIN TIME LOOP
```

This message can be ignored.

4.4.4 Program Operating Messages

When the program is initiated, the progress through the code is indicated on the screen. The following sequence will occur on the CRT indicating the program is progressing in a satisfactory manner.

```
* FLANK
- IN SOLVE 2 - AT BEGINNING
- IN SOLVE 2 - CALLING ASEMB2
- IN SOLVE 2 - ASEMB2 DONE, CALLING FORM B
- IN SOLVE 2 - DONE FORM B, CALLING APPTBC
- DONE APPTBC, CALLING ESQSOL2(0)
- FLOATING POINT - UNDERFLOW ERROR AT D70D2
- NEXT INSTRUCTION AT D70D8
```

This indicates that EQSOL2 is being executed which begins the matrix decomposition. The underflows in this section can be disregarded.

4.4.5 Time Step Indication

After the matrix has been decomposed and the temperature distributions at the various nodes are going to be calculated, the message

- BEGIN TIME STEP NO. 4 AT TIME 0.27000E + 04

occurs.

5. EXAMPLE PROBLEMS

The sample problems that follow are provided so that the user can see the input data as well as the output. In both cases, the input file is displayed but the complete output is displayed only for the static case since the output for the dynamic problem consists of about 200 computer pages. The output is obtained by typing PRINT FLANK.OUT on the PE. The output file - name is FLANK.OUT.

5.1 A STATIC PROBLEM

The input for this is shown as indicated on page A-90. Note that in the second line of FLANK.INP, the next to the last element is 0 (which sets ITRAN to 0). The output indicates that the net metering chamber energy is -114.7186 watts. The minus sign indicates that heat is flowing into the chamber which is reasonable since the MC temperature is 38°C whereas the EC temperature is 65°C. The heat transferred through the 1-D wall is 110.7266 watts. The two different frame coefficients are 2.62 percent and 3.48 percent.

5.2 A DYNAMIC PROBLEM

The input for the dynamic problem is also shown on page A-100. Notice that NUMIT.is - 1 indicating that a uniform initial temperature distribution given by TEMPØ is used. Line 16 provides this initial temperature (25°C). IPR is set equal to 4 so that the temperatures are printed out every 4th time step or in this case each hour. IQPR is the energy tally printout which is provided every two hours in this example since the time step is 15 minutes. After 24 hours, the frame coefficients were 0.48 percent and 0.95 percent.

6. REFERENCES

1. Lavine, A.G., Rucker, J.L., and Wilkes, K.E., "Flanking Loss Calibration for a Calibrated Hot Box," Owens-Corning Fiberglass, Technical Center, Granville, Ohio, (1981 or 1982).

2. Nakamura, S., "Computational Methods in Engineering and Science," Published by John Wiley & Sons, 1977, p. 143.

7. CODE LISTING

A listing of the code is provided so that definitions of various terms etc., may be obtained directly from the FORTRAN statements if desired. The code consists of a main section which controls the sequence of steps through the code which is determined by the input data.

The line numbers, right arrows and the message "! RANGE 2 NOT FOUND" are output from the editor and may be ignored.

```
C FLANK.FTN
C FLANKING LOSS CALCULATION FOR CALIBRATED HOT BOX
C IMPLICIT TIME INTEGRATION SCHEME,ALFA IS GREATER THAN ZERO
C   AND LESS THAN OR EQUAL TO ONE
C   LINEAR NODAL EQUATIONS,ALL COEFFICIENTS ARE CONSTANT
C CODED BY ONEGA AND BURNS ON 7/6/82
C
C  MATERIAL PROPERTIES:--------------------------------------------
C     MATERIAL NUMBER              MATERIAL TYPE
C
C        1                           POLYURETHANE FOAM
C        2                            FRP
C        3                           NEOPRENE RUBER GASKET
C        4                           INFLATABLE RUBBER GASKET
C        5                           WALL MATERIAL
C        6                           NULL MATERIAL
C
      COMMON /HEADG/ HEAD(16),TIMEM,DATEM
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TEMPBC/ NODEST(100),TBC(100)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
      COMMON /DBUG/ IDBUG(10)
      COMMON /HEAT/ Q(21),IEB,IQPR,ICHECK(5)
      COMMON /FIXT/ ITRAN
      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
      COMMON /PCONT/ ITITC(5,72),ITITXC(5,36),ITITYC(5,36),NC(5),
     *               NODESC(5,20)
      COMMON /PHIST/ ITITH(5,72),ITITXH(5,36),ITITYH(5,36),MH(5),
     *               NODESH(5,5),N,XMIN,XMAX
C
      DIMENSION C(246,246),A(246,246),UL(246,246)
      DIMENSION B(246),T(246),DT(246),EQUIL(246),IPVT(246),
     *  DX(246),RES(246),BTEMP(246)
C
      NNODES=246
C
C     DATA INPUT AND ECHO
C
      CALL INPUT2 (C,A,UL,B,T,DT,EQUIL,IPVT,DX,RES,NNODES,IDGT)
      IF (IDATA.NE.0) CALL ADIOS (1)
C
C     SOLUTION PHASE
C
      CALL SOLVE2 (C,A,UL,B,BTEMP,T,DT,EQUIL,IPVT,DX,RES,NNODES,
     *             IDGT)
C
C     WRITE TEMPERATURE FILE
C
      IF (NUMIT.LE.-2) WRITE (3,1001) (T(J),J=1,NUMNP)
C
      CALL ADIOS (1)
C
 1001 FORMAT (5E14.6)
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE ADIOS (N)
C
C*********************************************************************
C*     SUBROUTINE TO PERFORM EXIT FUNCTIONS FOR PROGRAM              *
C*********************************************************************
C
      CALL HEADER1
      IF (N.NE.2) GO TO 100
      WRITE (6,3000)
      GO TO 200
  100 WRITE (6,3001)
  200 CONTINUE
C
      STOP
C
 3000 FORMAT (///20X,"--- E R R O R   T E R M I N A T I O N ---"//)
 3001 FORMAT (///20X,"--- N O R M A L   T E R M I N A T I O N ---"
     *  ///)
C
      END
```

```
C
      SUBROUTINE HEADER1
C
C*********************************************************************
C*     SUBROUTINE TO PERFORM PAGING AND WRITE HEADER                 *
C*********************************************************************
C
      COMMON /HEADG/ HEAD(16),TIMEM,DATEM
      DATA STAR/4H****/
C
      WRITE (6,2000) (STAR,I=1,20)
      WRITE (6,2001) (HEAD(I),I=1,16)
      WRITE (6,2002) (STAR,I=1,20)
C
      RETURN
C
2000 FORMAT (1H1//20A4/)
2001 FORMAT (16A4)
2002 FORMAT (/20A4//)
C
      END
```

```
C
C*************************************************************************
C
      SUBROUTINE INPUT2 (C,A,UL,B,T,DT,EQUIL,IPVT,DX,RES,N,IDGT)
C
C*************************************************************************
C*     SUBROUTINE TO PREFORM DATA INPUT,INITIALIZATION AND ECHO         *
C*************************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /HEADG/ HEAD(16),TIMEM,DATEM
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TEMPBC/ NODEST(100),TBC(100)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
      COMMON /DBUG/ IDBUG(10)
      COMMON /HEAT/ Q(21),IEB,IQPR,ICHECK(5)
      COMMON /FIXT/ ITRAN
      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
C
      DIMENSION C(N,N),A(N,N),UL(N,N)
      DIMENSION B(N),T(N),DT(N),EQUIL(N),IPVT(N),DX(N),RES(N)
C
      READ (5,1000) (HEAD(I),I=1,16)
C
      CALL GEOM2 (IDGT)
C
      CALL GRIDBY
C
      CALL GRIDBZ
C
      CALL COEF
      DO 10 K=1,8
   10 TAMBP(K)=TAMB(K)
C
      CALL MAT
C
      IF (NUMTBC.NE.0) CALL TEMPB
C
C     TRANSIENT INPUT
C
      DELTA=1.
      ALFA=1.
      TIMO=0.
      TIMF=DELTA
      IF (ITRAN.NE.0) CALL TIMES
C
      IF (NUMIT.NE.0) CALL INIT (T,N,ITRAN)
C
      IF (IPLT.NE.0) CALL PLOTIN (ITRAN)
C
      IF (NUMNP.GT.N )GO TO 100
      CALL HEADER1
      WRITE (6,2000)
      RETURN
  100 WRITE (6,3000) NUMNP,N
      CALL ADIOS (2)
C
 1000 FORMAT (16A4)
 2000 FORMAT (//20X,"--- INPUT PHASE COMPLETE ---"//)
 3000 FORMAT (////20X,"NO. OF NODES POINTS  =",I10//
     * / 20X,"EXCEEDS DIMED VALUE    =",I10//
     * / 20X,"STOP ----------------.")
C
      END
```

```
C*********************************************************************
C
      SUBROUTINE GEOM2 (IDGT)
C
C*********************************************************************
C*     SUBROUTINE TO INPUT GRID                                      *
C*********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
      COMMON /DBUG/ IDBUG(10)
      COMMON /HEAT/ Q(21),IEB,IQPR,ICHECK(5)
      COMMON /FIXT/ ITRAN
      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
C
      READ (5,1000) NUMTBC,NUMIT,IDATA,IPR,IRFINE,IDGT,IEB,IQPR,
     *              ITRAN,IPLT
      READ (5,1000) (IDBUG(I),I=1,10)
      IF (IEB.NE.0) READ (5,1000) (ICHECK(I),I=1,5)
      READ (5,1000) NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      READ(5,1001) Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      READ(5,1000) NZPOLY,NZFRP,NZWALL
      READ(5,1001) ZPOLY,ZFRP,ZWALL
C
      CALL HEADER1
      WRITE (6,2000) NUMTBC,NUMIT,IDATA,IPR,IRFINE,IDGT,IEB,IQPR,
     *              ITRAN,IPLT
      WRITE (6,3000) (IDBUG(I),I=1,10)
      IF (IEB.NE.0) WRITE (6,3001) (ICHECK(I),I=1,5)
      CALL HEADER1
      WRITE (6,4000)
      WRITE (6,2001) NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      WRITE (6,2002) Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      WRITE (6,2003) NZPOLY,NZFRP,NZWALL
      WRITE (6,2004) ZPOLY,ZFRP,ZWALL
C
 1000 FORMAT (16I5)
 1001 FORMAT (8E10.0)
 2000 FORMAT (//
     * / 20X,"C O N T R O L   D A T A"//
     * / 20X,"NUMTBC                                =",I5/
     * / 20X,"NUMIT                                 =",I5/
     * / 20X,"IDATA                                 =",I5/
     * / 20X,"IPR                                   =",I5/
     * / 20X,"IRFINE                                =",I5/
     * / 20X,"IDGT                                  =",I5/
     * / 20X,"IEB                                   =",I5/
     * / 20X,"IQPR                                  =",I5/
     * / 20X,"ITRAN                                 =",I5/
     * / 20X,"IPLT                                  =",I5///)
 3000 FORMAT (//
     * / 20X,"DEBUG PARAMETERS"//
     * / 15X," 1 (WRITE A,C MATRICES, B VECTOR)  =",I5/
     * / 15X," 2 (WRITE DECOMPOSED C MATRIX   )  =",I5/
     * / 15X," 3 (WRITE NODAL ASSOC.          )  =",I5/
     * / 15X," 4 (WRITE ELEMENT MATRICES      )  =",I5/
     * / 15X," 5 (WRITE INTERIOR DATA         )  =",I5/
     * / 15X," 6 (WRITE C BEFORE DECOMP.      )  =",I5/
     * / 15X," 7 (WRITE B VECTOR              )  =",I5/
     * / 15X," 8 (                            )  =",I5/
     * / 15X," 9 (                            )  =",I5/
     * / 15X," 10(                            )  =",I5//)
 3001 FORMAT (//20X,"E N E R G Y   B A L A N C E S"/
     * / 15X,"ICHECK(1)                       =",I5/
     * / 15X,"ICHECK(2)                       =",I5/
     * / 15X,"ICHECK(3)                       =",I5/
     * / 15X,"ICHECK(4)                       =",I5/
     * / 15X,"ICHECK(5)                       =",I5)
 4000 FORMAT (//20X,"G R I D   S P E C I F I C A T I O N"//)
 2001 FORMAT (//20X,"Y NUMBERS FOLLOW"
     * / 20X,"NY5                                   =",I5/
     * / 20X,"NY4                                   =",I5/
     * / 20X,"NY1L                                  =",I5/
     * / 20X,"NYWALL                                =",I5/
     * / 20X,"NY1R                                  =",I5/
     * / 20X,"NY7                                   =",I5/
     * / 20X,"NY8                                   =",I5//)
 2002 FORMAT (/20X,"Y'S FOLLOW"/
     * / 20X,"Y5                                    =",E11.4/
     * / 20X,"Y4                                    =",E11.4/
     * / 20X,"Y1L                                   =",E11.4/
     * / 20X,"YWALL                                 =",E11.4/
     * / 20X,"Y1R                                   =",E11.4/
     * / 20X,"Y7                                    =",E11.4/
     * / 20X,"Y8                                    =",E11.4//)
 2003 FORMAT (/20X,"Z NUMBERS FOLLOW"/
     * / 20X,"NZPOLY                                =",I5/
     * / 20X,"NZFRP                                 =",I5/
     * / 20X,"NZWALL                                =",I5//)
 2004 FORMAT (/20X,"Z'S FOLLOW"/
     * / 20X,"ZPOLY                                 =",E11.4/
     * / 20X,"ZFRP                                  =",E11.4/
     * / 20X,"ZWALL                                 =",E11.4//)
C
       RETURN
```

```
C*********************************************************************
C
      SUBROUTINE GRIDBY
C
C*********************************************************************
C*   SUBROUTINE TO SPECIFY ALL THE Y BOURDARY NODE NUMBERS          *
C*********************************************************************
C
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
C
       NTY4=NY5+NY4
       NTY1L=NTY4+NY1L
       NTYWAL=NTY1L+NYWALL
       NTY1R=NTYWAL+NY1R
       NTY7=NTY1R+NY7
       NCOL=NTY7+NY8
C
       RETURN
C
       END
```

```
C*********************************************************************
C
      SUBROUTINE GRIDBZ
C
C*********************************************************************
C*   SUBROUTINE TO SPECIFY BOUNDARY NODES IN THE Z DIRECTION        *
C*********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
C
       NTZFRP=NZPOLY+NZFRP
       NTZWAL=NTZFRP+NZWALL
      NUMNP=NCOL*NTZFRP+(NYWALL+1)*NZWALL
      NUMNPL=NTZFRP*NCOL
C
      CALL HEADER1
      WRITE (6,2000)
      WRITE (6,2001) NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      WRITE (6,2002) NTZFRP,NTZWAL
      WRITE (6,2003) NUMNP,NUMNPL
C
       RETURN
C
 2000 FORMAT (
     * / 25X,"C U M U L A T I V E   N O D E   N U M B E R S"//)
 2001 FORMAT (/20X,"Y - DIRECTION NODE NUMBERS"/
     * / 20X,"NTY4                                 =",I5/
     * / 20X,"NTY1L                                =",I5/
     * / 20X,"NTYWAL                               =",I5/
     * / 20X,"NTY1R                                =",I5/
     * / 20X,"NTY7                                 =",I5/
     * / 20X,"NCOL                                 =",I5)
 2002 FORMAT (/20X,"Z - DIRECTION NODE NUMBERS"/
     * / 20X,"NTZFRP                               =",I5/
     * / 20X,"NTZWAL                               =",I5/)
 2003 FORMAT (/20X,"TOTAL NUMBER OF NODES (EQUATIONS) =",I10/
     * / 20X,"LOWER NUMBER OF NODE POINTS        =",I10///)
C
       END
```

```
C***********************************************************************
C
      SUBROUTINE INIT (T,N,ITRAN)
C
C***********************************************************************
C*     SUBROUTINE TO READ IN AND ECHO INITIAL TEMPERATURES            *
C***********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
C
      DIMENSION T(N)
C
      IF (ITRAN.EQ.0) GO TO 50
      IF (NUMIT.GT.0) GO TO 200
      READ (5,1001) TEMPO
C
C      CORRECT FOR STEADY CALCS.
C
   50 IF (ITRAN.EQ.0) TEMPO=0.
C
      DO 100 I=1,N
  100 T(I)=TEMPO
C
      CALL HEADER1
      WRITE (6,2000) TEMPO
C
      RETURN
C
  200 CONTINUE
      IF (NUMIT.GE.2) GO TO 300
      READ (5,1003) (NODE,T(NODE),I=1,NUMNP)
      GO TO 400
  300 READ (1,1002) (T(I),I=1,NUMNP)
C
  400 CALL HEADER1
      WRITE (6,2000)
      WRITE (6,2001) (T(NODE),NODE=1,NUMNP)
C
      RETURN
C
 1001 FORMAT (8E10.0)
 1002 FORMAT (5E14.0)
 1003 FORMAT (5(I5,E10.0))
 2000 FORMAT (///20X,"I N I T I A L   T E M P E R A T U R E S"//
     * / 15X,"INITIAL TEMPERATURE                          =",E14.5)
 2001 FORMAT (8(E10.0,4X))
C
      END

C***********************************************************************
C
      SUBROUTINE MAT
C
C***********************************************************************
C*     SUBROUTINE TO INPUT AND ECHO MATERIAL PROPERTIES               *
C***********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
C
      READ (5,1001) (RKAY(I),I=1,5)
      READ (5,1001) (RHO(I),I=1,5)
      READ (5,1001) (CP(I),I=1,5)
      RKAY(6)=0.
      RHO(6)=0.
      CP(6)=0.
C
      CALL HEADER1
      WRITE (6,2000)
      DO 100 J=1,6
  100 WRITE (6,2001) J,RKAY(J),RHO(J),CP(J)
C
      DO 200 J=1,5
  200 RHO(J)=RHO(J)*CP(J)
C
      RETURN
C
 1001 FORMAT (8E10.0)
 2000 FORMAT ( //25X,"M A T E R I A L   P R O P E R T I E S"
     * /// 17X,"I",10X,"RKAY",13X,"RHO",14X,"CP"//)
 2001 FORMAT (15X,I5,3(5X,E11.4))
C
      END
```

```
C***********************************************************************
C
      SUBROUTINE TEMPB
C
C***********************************************************************
C*    SUBROUTINE TO READ TEMPERATURE BCS                               *
C***********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /TEMPBC/ NODEST(100),TBC(100)
C
      READ (5,1003) (NODEST(I),TBC(I),I=1,NUMTBC)
C
      CALL HEADER1
      WRITE (6,2000) NUMTBC
      WRITE (6,2001) (NODEST(I),TBC(I),I=1,NUMTBC)
C
      RETURN
C
 1003 FORMAT (5(I5,E10.0))
 2000 FORMAT (1H1
     * / 25X,"T E M P E R A T U R E   B O U N D A R Y   C O N D"
     * / 40X,"NUMBER OF TEMPERATURE BCS      =",I5//)
 2001 FORMAT (5(3X,I5,2X,E10.3))
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE TIMES
C
C***********************************************************************
C*    SUBROUTINE TO INPUT AND ECHO TIMING INFORMATION                  *
C***********************************************************************
C
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      READ (5,1001) TIMO,TIMF,DELTA,ALFA
C
      CALL HEADER1
      WRITE (6,2000)
      WRITE (6,2001) TIMO,TIMF,DELTA,ALFA
C
      RETURN
C
 1001 FORMAT (8E10.0)
 2000 FORMAT (//25X,"T I M I N G   P A R A M E T E R S"//)
 2001 FORMAT (/20X,"TIMES, ETC."
     * / 20X,"TIMO                                          =",E11.4/
     * / 20X,"TIMF                                          =",E11.4/
     * / 20X,"DELTA                                         =",E11.4/
     * / 20X,"ALFA                                          =",E11.4/)
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE SOLVE2 (C,A,UL,B,BTEMP,T,DT,EQUIL,IPVT,DX,RES,N,
     *                   IDOT)
C
C***********************************************************************
C*     SUBROUTINE TO PERFORM TIME INTEGRATION AND EBS                  *
C***********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /HEADG/ HEAD(16),TIMEM,DATEM
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TEMPBC/ NODEST(100),TBC(100)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
      COMMON /DBUG/ IDBUG(10)
      COMMON /HEAT/ Q(21),IEB,IQPR,ICHECK(5)
      COMMON /FIXT/ ITRAN
      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
      COMMON /PCONT/ ITITC(5,72),ITITXC(5,36),ITITYC(5,36),NC(5),
     *               NODESC(5,20)
      COMMON /PHIST/ ITITH(5,72),ITITXH(5,36),ITITYH(5,36),MH(5),
     *               NODESH(5,5),NH,XMIN,XMAX
C
      DIMENSION C(N,N),A(N,N),UL(N,N)
      DIMENSION B(N),BTEMP(N),T(N),DT(N),EQUIL(N),IPVT(N),DX(N),
     *          RES(N)
C
      DATA BIG /1.E30/
      WRITE (7,7000)
 7000 FORMAT(" IN SOLVE2 - AT BEGINNING")
C
C      ASSEMBLE A AND C MATRICES AND THE B VECTOR
C
      DO 100 J=1,N
      BTEMP(J)=0.0
      B(J)=0.
      DT(J)=0.
      DO 100 I=1,N
      C(I,J)=0.
  100 A(I,J)=0.
      WRITE (7,7001)
 7001 FORMAT (" IN SOLVE2 - CALLING ASEMB2")
      CALL ASEMB2 (C,A,N)
      WRITE (7,7002)
 7002 FORMAT (" IN SOLVE2 - ASEMB2 DONE, CALLING FORMB")
      CALL FORMB (0,B,BTEMP,N)
C
C      FORM AND REDUCE SOLUTION MATRIX
C
      CDIV=1.
      IF (ITRAN.EQ.0) CDIV=1.E30
      DO 200 J=1,NUMNP
      DO 200 I=1,NUMNP
  200 C(I,J)=C(I,J)/CDIV + A(I,J)
C
C      ADD TEMP BCS FOR STEADY STATE PROBLEM
C
      IF (NUMTBC.EQ.0) GO TO 220
      DO 210 I=1,NUMTBC
      J=NODEST(I)
  210 C(J,J)=BIG
      WRITE (7,7003)
 7003 FORMAT (" IN SOLVE2 - DONE FORMB, CALLING APPTBC")
      IF (ITRAN.EQ.0) CALL APPTBC (NUMTBC,ITRAN,BIG,B,T,N)
  220 CONTINUE
C
C      WRITE OUT DBUG MATRICES
C
      IF (IDBUG(1).EQ.0) GO TO 260
      WRITE (6,3000)
      DO 251 I=1,NUMNP
      WRITE (6,4001) I
  251 WRITE (6,4000) (A(I,J),J=1,NUMNP)
      WRITE (6,3001)
      DO 255 I=1,NUMNP
      WRITE (6,4001) I
  255 WRITE (6,4000) (C(I,J),J=1,NUMNP)
      WRITE (6,3002)
      WRITE (6,4000) (B(J),J=1,NUMNP)
      IF (IDBUG(1).LT.0) CALL ADIOS (1)
  260 CONTINUE
C
```

```
      WRITE (7,7004)
 7004 FORMAT (" IN SOLVE2 - DONE APTBC, CALLING EQSOL2(0)")
      CALL EQSOL2 (0,C,A,UL,B,T,DT,EQUIL,IPVT,DX,RES,N,IDGT,
     *             TIME)
C
      IF (IDBUG(2).LT.0) CALL ADIOS (1)
C
C      WRITE INITIAL STATE INTO PLOT FILE
C
      IF (IPLT.EQ.0) GO TO 270
      IF (NUMCON.NE.0) CALL PLOTC (0,TIME,T,N)
      IF (NUMHIS.NE.0) CALL PLOTH (0,TIME,T,N)
C
C      ICOUNT - COUNTER FOR PRINTING
C      KOUNT  - COUNTER FOR ENERGY BALANCE WRITES
C      KOUNTC - COUNTER FOR CONTOUR PLOTS
C      KOUNTH - COUNTER FOR TIME HISTORY PLOTS
C
  270 ISTEP=0
      ICOUNT=0
      KOUNT=0
      KOUNTC=0
      KOUNTH=0
      TIME=TIMO
C
C      PRINT INITIAL TEMPERATURES
C
      CALL PRINT2 (TIME,T,N,NUMNP)
C
C      TIME STEP BEGINS
C
      WRITE (7,7005)
 7005 FORMAT (" IN SOLVE2 - DONE DECOMPOSITION , BEGIN TIME LOOP")
  300 ISTEP=ISTEP+1
      WRITE (7,7006) ISTEP,TIME
 7006 FORMAT(" BEGIN TIME STEP NO. ",I5,"  AT TIME ",E12.5)
      ICOUNT=ICOUNT+1
      KOUNT=KOUNT+1
      ALTIM=TIME+ALFA*DELTA
      TIME=TIME+DELTA
C
C      FORM RHS - B VECTOR
C
      IF (ITRAN.GE.0) GO TO 350
      READ (5,1001) (TBC(I),I=1,NUMTBC)
      READ (5,1001) (TAMBP(I),I=1,8)
  350 IF (ITRAN.NE.0) CALL FORMB (ITRAN,B,BTEMP,N)
      IF (ITRAN.NE.0) CALL LODE2 (B,T,A,N,ALFA)
      IF (ITRAN.NE.0) CALL APPTBC (NUMTBC,ITRAN,BIG,B,T,N)
C
C      SOLUTION PHASE
C
      CALL EQSOL2 (1,C,A,UL,B,T,DT,EQUIL,IPVT,DX,RES,N,IDGT,
     *             TIME)
C
C      UPDATE TEMPERATURES
C
      CALL UPDATE (T,DT,N)
C
C      PRINTING
C
      IF (ICOUNT.LT.IPR) GO TO 400
      CALL PRINT2 (TIME,T,N,NUMNP)
      ICOUNT=0
C
C      PERFORM ENERGY BALANCES AND PRINTING THEREOF
C
  400 IF (IEB.EQ.0) GO TO 500
      CALL QSEMB2 (T,N)
      CALL QSOLV2 (ISTEP,KOUNT,T,N,TIME)
C
C      PLOTTING
C
  500 IF (IPLT.EQ.0) GO TO 600
C
C      CONTOURS
C
      IF (KOUNTC.LT.NSTEPC) GO TO 550
      IF (NUMCON.NE.0) CALL PLOTC (0,TIME,T,N)
      KOUNTC=0
C
C      TIME HISTORIES
C
  550 IF (KOUNTH.LT.NSTEPH) GO TO 600
      IF (NUMHIS.NE.0) CALL PLOTH (0,TIME,T,N)
      KOUNTH=0
C
C      TEST FOR END OF TIME LOOP
C
```

```
 600 IF (TIME.LT.TIMF) GO TO 300
C
C     FINAL PLOTS
C
      IF (NUMCON.NE.0) CALL PLOTC (1,TIME,T,N)
      IF (NUMHIS.NE.0) CALL PLOTH (0,TIME,T,N)
      IF (NUMHIS.NE.0) CALL PLOTH (1,TIME,T,N)
C
      RETURN
C
1001 FORMAT (8E10.0)
3000 FORMAT (////
    * / 15X,"A MATRIX FOLLOWS"//)
3001 FORMAT (////
    * / 15X,"C MATRIX FOLLOWS"//)
3002 FORMAT (////
    * / 15X,"B VECTOR FOLLOWS"//)
4000 FORMAT (10(1X,E11.4))
4001 FORMAT ( // 25X,"ROW NUMBER     -"I5//)
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE ACONV (N,M,DY,DZ,MM,A0,A1,A2,A3,A4,C0)
C
C********************************************************************
C*     SUBROUTINE TO DETERMINE A(IJ) FOR CONVECTION BOUNDARY        *
C*********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=(RHO(MM)*DY*DZ / (2.*DELTA))
      A01=HTC(M)*DY
      A1=-RKAY(MM)*DZ/(2.*DY)
      A2=A1
      A3=-RKAY(MM)*DY/DZ
      A4=A3
C
      GO TO (1,2,3,4) ,N
    1 A3=A2
      A2=A4
      A4=A1
      A1=0.
      GO TO 10
C
    2 A1=A3
      A3=A2
      A4=A3
      A2=0.
      GO TO 10
C
    3 A3=0.
      GO TO 10
C
    4 A4=0.
   10 A0=-(A1+A2+A3+A4)+A01
C
      RETURN
C
      END


C********************************************************************
C
      SUBROUTINE AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,
     *                 A0,A1,A2,A3,A4,C0)
C
C********************************************************************
C*     SUBROUTINE TO CALCULATE A(I,J) FOR INTERIOR CORNERS          *
C********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=(RHO(MA)*DX2*DX4 + RHO(MB)*DX4*DX1 + RHO(MC)*DX1*DX3 +
     *    RHO(MD)*DX3*DX2) / (4.*DELTA)
      A0=(RKAY(MA)*(DX2/DX4+DX4/DX2) + RKAY(MB)*(DX1/DX4+DX4/DX1) +
     *    RKAY(MC)*(DX3/DX1+DX1/DX3) + RKAY(MD)*(DX2/DX3+DX3/DX2))
     *    / 2.
      A1=-(RKAY(MB)*DX4/DX1 + RKAY(MC)*DX3/DX1) / 2.
      A2=-(RKAY(MD)*DX3/DX2 + RKAY(MA)*DX4/DX2) / 2.
      A3=-(RKAY(MC)*DX1/DX3 + RKAY(MD)*DX2/DX3) / 2.
      A4=-(RKAY(MA)*DX2/DX4 + RKAY(MB)*DX1/DX4) / 2.
C
      RETURN
C
      END
```

```
C
C**********************************************************************
C
      SUBROUTINE AIC2 (N1,N2,NH1,NH2,MA,MB,MC,MD,
     *                 DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
C
C***********************************************************************
C*     SUBROUTINE TO CALCULATE A(IJ) FOR REENTRANT CORNERS             *
C***********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=(RHO(MA)*DX4*DX2 + RHO(MB)*DX1*DX4 +RHO(MC)*DX1*DX3 +
     *     RHO(MD)*DX3*DX2) / (4.*DELTA)
      DDX1=DX1
      DDX2=DX4
      IF (N1.EQ.2) DDX1=DX2
      A0=0.5*HTC(NH1)*DDX1
      A01=0.5*HTC(NH2)*DDX2
      A1=-(RKAY(MB)*DX4/DX1 + RKAY(MC)*DX3/DX1) / 2.
      A2=-(RKAY(MD)*DX3/DX2 + RKAY(MA)*DX4/DX2) / 2.
      A3=-(RKAY(MC)*DX1/DX3 + RKAY(MD)*DX2/DX3) / 2.
      A4=-(RKAY(MA)*DX2/DX4 + RKAY(MB)*DX1/DX4) / 2.
C
      A0=-(A1+A2+A3+A4)+A0+A01
C
      RETURN
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE AIC3 (N,NH,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
C
C***********************************************************************
C*     SUBROUTINE TO COMPUTE A(IJ)FOR CORNER NODE WITH CONVECTION      *
C*       ON ONE SIDE                                                   *
C***********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=(RHO(MA)*DXA + RHO(MB)*DXB)*DX/(4.*DELTA)
      A01=HTC(NH)*0.5*(DXA+DXB)
      A1=-(RKAY(MA)*DXA + RKAY(MB)*DXB) / (2.*DX)
      A2=A1
      A3=-RKAY(MA)*DX/(2.*DXA)
      A4=-RKAY(MB)*DX/(2.*DXB)
      GO TO (1,2,3,4) ,N
    1 A1=0.
      GO TO 10
C
    2 A2=A3
      A3=A4
      A4=A2
      A2=0.
      GO TO 10
C
    3 A1=A4
      A4=A2
      A2=A3
      A3=0.
C
      GO TO 10
    4 A1=A3
      A3=A2
      A2=A4
      A4=0.
C
   10 A0=-(A1+A2+A3+A4)+A01
C
      RETURN
C
      END
```

```
C
C*******************************************************************
C
      SUBROUTINE AIC4 (N1,N2,NH1,NH2,M,DX1,DX2,
     *        A0,A1,A2,A3,A4,C0)
C
C********************************************************************
C*    SUBROUTINE TO CALCULATE A(I,J) FOR EXTERIOR CORNER           *
C*        WITH HEAT TRANSFER ON TWO SIDES                          *
C********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=RHO(M)*DX1*DX2/(4.*DELTA)
      A1=-RKAY(M)*DX2/(2.*DX1)
      A2=A1
      A3=-RKAY(M)*DX1 / (2.*DX2)
      A4=A3
      A0=0.5*DX2*HTC(NH1)
      A01=0.5*DX1*HTC(NH2)
      IF (N1.NE.1) GO TO 2
      IF (N2.NE.3) GO TO 1
C
      A1=0.
      A3=0.
      GO TO 10
C
    1 A1=0.
      A4=0.
      GO TO 10
C
    2 IF (N2.NE.3) GO TO 3
      A2=0.
      A3=0.
      GO TO 10
C
    3 A2=0.
      A4=0.
   10 A0=A0+A01-(A1+A2+A3+A4)
C
      RETURN
C
      END


C
C*********************************************************************
C
     SUBROUTINE AINF (MM,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
C
C********************************************************************
C*    SUBROUTINE TO CALCULATE A(IJ) AT THE INTERFACES              *
C********************************************************************
C
     COMMON /MATS/ RKAY(6),RHO(6),CP(6)
     COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
     C0=(RHO(MA)*DXA + RHO(MB)*DXB)*DX / (2.*DELTA)
     A1=-(RKAY(MA)*DXA/(2.*DX) + RKAY(MB)*DXB/(2.*DX))
     A2=A1
     A3=-RKAY(MB)*DX/DXB
     A4=-RKAY(MA)*DX/DXA
     A0=-(A1+A2+A3+A4)
C
     IF (MM.EQ.1) RETURN
     A1=A4
     A4=A2
     A2=A3
     A3=A4
C
     RETURN
C
     END
```

```
C*********************************************************************
C
      SUBROUTINE AINT (DY,DZ,M,A0,A1,A2,A3,A4,C0)
C
C**********************************************************************
C*    ROUTINE TO CALCULATE A(IJ) FOR GENERAL INTERIOR NODE           *
C**********************************************************************
C
      COMMON /DBUG/ IDBUG(10)
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      C0=(RHO(M)*DY*DZ/DELTA)
      A0=RKAY(M)*2.*(DY/DZ+DZ/DY)
      A1=-RKAY(M)*DZ/DY
      A2=A1
      A3=-RKAY(M)*DY/DZ
      A4=A3
C
      IF (IDBUG(5).EQ.0) RETURN
      WRITE (6,3000) M,RKAY(M),C0,A0,A1,A2,A3,A4
C
      RETURN
C
 3000 FORMAT (5X,"M,K,C0,A(5) =",I5,7(2X,E12.5))
C
      END


C
C**********************************************************************
C
      SUBROUTINE AMATRX (A,N,NN,AN)
C
C**********************************************************************
C*    SUBROUTINE TO PERFORM ALGEBRAIC ASSEMBLY OF THE A MATRIX       *
C**********************************************************************
C
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION NN(5),AN(5)
      DIMENSION A(N,N)
C
      DO 100 K=1,5
  100 A(NN(1),NN(K))=A(NN(1),NN(K))+AN(K)
C
      IF (IDBUG(4).EQ.0) RETURN
      WRITE (6,3000) (NN(L),L=1,5),(AN(L),L=1,5)
C
      RETURN
C
 3000 FORMAT (5X,"IN AMATRX - NN(5),AN(5) =",2X,5I5,5(2X,E12.5))
C
      END


C
C**********************************************************************
C
      SUBROUTINE APPTBC (NUMTBC,ITRAN,BIG,B,T,NN)
C
C**********************************************************************
C*    SUBROUITNE TO APPLY TEMPERATURE BCS VIA PENALTY FUNCTION       *
C**********************************************************************
C
      COMMON /TEMPBC/ NODEST(100),TBC(100)
C
      DIMENSION B(NN),T(NN)
C
      IF (NUMTBC.EQ.0) RETURN
C
      DO 100 I=1,NUMTBC
      J=NODEST(I)
  100 B(J)=BIG*(TBC(I)-T(J))
C
      RETURN
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE EQSOL2 (K,C,A,UL,B,T,DT,EQUIL,IPVT,DX,RES,N,IDGT,
     *                    TIME)
C
C***********************************************************************
C     SUBROUTINE TO PERFORM MATRIX CALCULATIONS                        *
C***********************************************************************
C
      COMMON /CONTRL/ NUMNP,NUMIT,NUMNPL,NUMTBC,IPR,IDATA,TEMPO
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION C(N,N),A(N,N),UL(N,N)
      DIMENSION B(N),T(N),DT(N),EQUIL(N),IPVT(N),DX(N),RES(N)
C
      IF (K) 200,100,200
C
C     LU DECOMPOSITION
C
  100 CONTINUE
      IA=N
      NN=NUMNP
      IF (IDBUG(6).EQ.0) GO TO 120
      WRITE (6,3500)
      DO 110 I=1,NUMNP
      WRITE (6,3501) I
  110 WRITE (6,3502) (C(I,J),J=1,NUMNP)
      IF (IDBUG(6).LT.0) CALL ADIOS (1)
  120 CONTINUE
      CALL LUDATF (C,UL,NN,IA,IDGT,D1,D2,IPVT,EQUIL,WA,IER)
      D=D1*2.**D2
      CALL HEADER1
      WRITE (6,2000) D,WA,NN,IA,IER
      IF (IER.EQ.0) GO TO 150
      WRITE (6,3000) IER
      IF (IER.GE.128) CALL ADIOS (2)
  150 IF (IDBUG(2).EQ.0) RETURN
      CALL HEADER1
      WRITE (6,4000)
C
      DO 170 I=1,NUMNP
      WRITE (6,3501) I
  170 WRITE (6,3502) (C(I,J),J=1,NUMNP)
      IF (IDBUG(2).LT.0) CALL ADIOS (1)
C
      RETURN
C
C     SOLUTION PHASE
C
  200 CONTINUE
      CALL LUELMF (UL,B,IPVT,NN,IA,DT)
C
      IF (IDBUG(7).EQ.0) GO TO 300
C
C     WRITE SOLUTION VECTOR
C
      WRITE (6,5000) TIME
      WRITE (6,3502) (B(J),J=1,NUMNP)
  300 CONTINUE
C
C     REFINEMENT
C
      IF (IRFINE.EQ.0) RETURN
C
      CALL LUREFF (C,B,UL,IPVT,NN,IA,DT,IDGT,RES,DX,IER)
      IF (IER.EQ.0) RETURN
      WRITE (6,3001) IER
      CALL ADIOS (2)
C
 2000 FORMAT (//
     * / 15X,"E Q U A T I O N   D A T A"//
     * / 15X,"DETERMINANT                            =",E14.7/
     * / 15X,"ACCURACY TEST PARAMETER                =",E14.7/
     * / 15X,"NUMBER OF EQUATIONS                    =",I10/
     * / 15X,"ROW SIZE IN CALLING ROUTINE            =",I10/
     * / 15X,"ERROR PARAMETER (129-SIN.,34-ACC.) =",I10///)
 3000 FORMAT (//
     * / 15X,"--- ERROR FROM EQUATION SOLVER ---"/
     * / 20X,"DECOMPOSITION PHASE"/
     * / 20X,"IER   =",I10//)
 3001 FORMAT (//
     * / 15X,"--- ERROR FROM EQUATION SOLVER ---"/
     * / 20X,"REFINEMENT PHASE"/
     * / 20X,"IER   =",I10//)
 3500 FORMAT (//
     * 15X,"IN EQSOL2 --- C MATRIX BEFORE DECOMPOSITION FOLLOWS"//
     * )
 3501 FORMAT (//25X,"R O W    N U M B E R     =",I10//)
 3502 FORMAT (10(2X,E11.4))
 4000 FORMAT (//15X,"IN EQSOL2 --------------"/
     * / 15X,"DECOMPOSED CAPACITANCE MATRIX FOLLOWS"///)
 5000 FORMAT (//
     * / 25X,"--- IN EQSOL2   --- AT TIME    =",E14.7//
     * / 30X,"R H S   V E C T O R"//)
C
      END
```

```
C**********************************************************************
C
      SUBROUTINE GETNOD (I,J,N)
C
C**********************************************************************
C*    SUBROUTINE TO MAP FROM I,J  TO NODE NUMBER N(5)                *
C**********************************************************************
C
      COMMON/CONTRL/NUMNP,NUMIT,NUMNPL,NUMTBC,IPR.IDATA,TEMPO
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION N(5)
C
      IF (I.GT.NTZFRP+1) GO TO 300
      IF (I.GE.NTZFRP) GO TO 200
C
C     LOWER REGION
C
      N(1)=(I-1)*NCOL+J
      N(2)=N(1)-1
      N(3)=N(1)+1
      N(4)=N(1)-NCOL
      N(5)=N(1)+NCOL
      N(2)=MAXO(N(2),1)
      N(4)=MAXO(N(4),1)
      N(3)=MINO(N(3),NUMNPL)
      N(5)=MINO(N(5),NUMNPL)
C
      GO TO 400
C
C     MIDDLE REGION
C
  200 IF (I.GT.NTZFRP) GO TO 250
C
C     ON TOP OF BASE
C
      N(1)=(I-1)*NCOL+J
      N(2)=N(1)-1
      N(3)=N(1)+1
      N(4)=N(1)-NCOL
      N(5)=NUMNPL+(J-NTY1L+1)
      N(5)=MINO(N(5),NUMNP)
      GO TO 400
C
C     IN UPPER WALL
C
  250 N(1)=NUMNPL+(J-NTY1L+1)
      N(2)=N(1)-1
      N(3)=N(1)+1
      N(4)=NUMNPL-(NCOL-J)
      N(5)=N(1)+(NTYWAL-NTY1L+1)
      GO TO 400
C
C   UPPER REGION
C
  300 N(1)=NUMNPL+(I-NTZFRP-1)*(NTYWAL-NTY1L+1)+(J-NTY1L+1)
      N(2)=N(1)-1
      N(3)=N(1)+1
      N(4)=N(1)-(NTYWAL-NTY1L+1)
      N(5)=N(1)+(NTYWAL-NTY1L+1)
      N(3)=MINO(N(3),NUMNP)
      N(5)=MINO(N(5),NUMNP)
C
  400 IF (IDBUG(3).EQ.0) RETURN
      WRITE (6,3000) I,J.N(1),N(2),N(3),N(4),N(5)
C
 3000 FORMAT (15X,"IN GETNOD - I,J,N(5)  =",8I5)
C
      END
```

```
C***********************************************************************
C
      SUBROUTINE PRINT2 (TIME,T,N,NUMNP)
C
C***********************************************************************
C*      SUBROUTINE TO PRINT TEMPERATURE STATES                         *
C***********************************************************************
C
      DIMENSION T(N),L(5),U(5)
C
      N5=NUMNP/5
      NLEFT=NUMNP-5*N5
      NDO5=N5
      IF (NLEFT.NE.0) NDO5=N5+NLEFT-4
      NSTART=NDO5+1
      NSTOP=NSTART+4-NLEFT
C
      CALL HEADER1
      WRITE (6,2000) TIME
C
      DO 200 I=1,NDO5
      DO 100 J=1,5
      L(J)=(J-1)*NSTOP+I
  100 U(J)=T(L(J))
  200 WRITE(6,2001)L(1),U(1),L(2),U(2),L(3),U(3),L(4),U(4),L(5),U(5)
C
      IF (NLEFT.EQ.0) RETURN
C
      DO 300 I=NSTART,NSTOP
      DO 350 J=1,4
      L(J)=(J-1)*NSTOP+I
  350 U(J)=T(L(J))
  300 WRITE(6,2001)L(1),U(1),L(2),U(2),L(3),U(3),L(4),U(4)
C
 2000 FORMAT (///
     * / 15X,"T E M P E R A T U R E   S T A T E"/
     * / 15X,"  A T   T I M E   = ",E12.5////)
 2001 FORMAT (5(2X,I5,3X,E15.7))
C
      RETURN
C
      END


C***********************************************************************
C
      SUBROUTINE LODE2 (B,T,A,N,ALFA)
C
C***********************************************************************
C*      SUBROUTINE TO FORN RHS DURING TRANSIENT RUN                    *
C***********************************************************************
C
      DIMENSION B(N),T(N),A(N,N)
C
      DO 200 I=1,N
      SUM=0.
      DO 100 J=1,N
  100 SUM=SUM+A(I,J)*T(J)
  200 B(I)=B(I)-SUM
C
      RETURN
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE UPDATE (T,DT,N)
C
C*********************************************************************
C*     SUBROUTINE TO UPDATE TEMPERATURES                             *
C*********************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
C
      DIMENSION T(N),DT(N)
C
      DO 100 I=1,N
  100 T(I)=T(I)+DT(I)
C
      DO 200 I=1,8
  200 TAMB(I)=TAMBP(I)
C
      RETURN
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE WPT (A,C,N,I,J,A0,A1,A2,A3,A4,C0)
C
C*********************************************************************
C*    SUBROUTINE TO WRITE A POINT TO THE A MATRIX                    *
C*********************************************************************
C
C
      DIMENSION NN(5),AN(5)
      DIMENSION A(N,N),C(N,N)
C
      AN(1)=A0
      AN(2)=A1
      AN(3)=A2
      AN(4)=A3
      AN(5)=A4
C
      CALL GETNOD (I,J,NN)
      CALL AMATRX (A,N,NN,AN)
      C(NN(1),NN(1))=C(NN(1),NN(1))+C0
C
      RETURN
C
      END
```

```
  *******************************************************************

      SUBROUTINE WMAT1 (A,C,N,I,NC1,NC2,A0,A1,A2,A3,A4,C0)

  *******************************************************************


      DIMENSION AN(5),NN(5)
      DIMENSION A(N,N),C(N,N)

      AN(1)=A0
      AN(2)=A1
      AN(3)=A2
      AN(4)=A3
      AN(5)=A4

      DO 100 J=NC1,NC2
      CALL GETNOD (I,J,NN)
      CALL AMATRX (A,N,NN,AN)
      C(NN(1),NN(1))=C(NN(1),NN(1))+C0
  100 CONTINUE

      RETURN

      END
```

```
C
C***********************************************************************
C
      SUBROUTINE WMAT2 (A,C,N,NR1,NR2,J,A0,A1,A2,A3,A4,
     *                  C0)
C
C***********************************************************************
C*    SUBROUTINE TO LOGICALLY ASSEMBLE A ROW OF THE A MATRIX          *
C***********************************************************************
C
C
      DIMENSION AN(5),NN(5)
      DIMENSION A(N,N),C(N,N)
C
      AN(1)=A0
      AN(2)=A1
      AN(3)=A2
      AN(4)=A3
      AN(5)=A4
C
      DO 100 I=NR1,NR2
      CALL GETNOD (I,J,NN)
      CALL AMATRX (A,N,NN,AN)
      C(NN(1),NN(1))=C(NN(1),NN(1))+C0
  100 CONTINUE
C
      RETURN
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE WRTMAT (A,C,N,NR1,NR2,NC1,NC2,A0,A1,A2,
     *                   A3,A4,C0)
C
C***********************************************************************
C*    SUBROUTINE TO WRITE ROWS AND COLUMNS OF GEOMETRY TO             *
C*    THE A MATRIX                                                    *
C***********************************************************************
C
C
      DIMENSION NN(5),AN(5)
      DIMENSION A(N,N),C(N,N)
C
      AN(1)=A0
      AN(2)=A1
      AN(3)=A2
      AN(4)=A3
      AN(5)=A4
C
      DO 100 I=NR1,NR2
      DO 100 J=NC1,NC2
      CALL GETNOD (I,J,NN)
      CALL AMATRX (A,N,NN,AN)
  100 CONTINUE
C
      RETURN
C
      END
```

```
$BATCH
C
C***********************************************************************
C
      SUBROUTINE ASEMB2 (C,A,NN)
C
C***********************************************************************
C*  SUBROUTINE TO FORM ELEMENTS OF THE A MATRIX                         *
C***********************************************************************
C
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION A(NN,NN),C(NN,NN)
C
C     1. INTERIOR NODE POINTS
C
C       - REGION 5.
C
      DY=Y5/FLOAT(NY5-1)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NC1=2
      NC2=NY5-1
      NR1=2
      NR2=NZPOLY-1
      CALL AINT (DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 4L
C
      DY=(Y4-Y5)/FLOAT(NY4)
      DZ=ZPOLY/(NZPOLY-1)
      NC1=NY5+1
      NC2=NTY4-1
      CALL AINT (DY,DZ,3,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 4U
C
      DZ=(ZFRP-ZPOLY)/NZFRP
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL AINT (DY,DZ,3,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 1L
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NR2=NZPOLY-1
      NC1=NTY4+1
      NC2=NTY1L-1
      NR1=2
      CALL AINT (DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 1C
C
```

```
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NC1=NTY1L+1
      NC2=NTYWAL-1
      NR1=2
      NR2=NZPOLY-1
      CALL AINT (DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 1R
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL AINT (DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 7L
C
      DY=(Y7-Y1R)/FLOAT(NY7)
      NC1=NTY1R+1
      NC2=NTY7-1
      NR2=NZPOLY-1
      CALL AINT (DY,DZ,4,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 7U
C
      DZ=(ZFRP-ZPOLY)/NZFRP
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL AINT (DY,DZ,4,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 8
C
      DY=(Y8-Y7)/FLOAT(NY8)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NC1=NTY7+1
      NC2=NCOL-1
      NR1=2
      NR2=NZPOLY-1
      CALL AINT (DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 6
C
      DY=Y5/FLOAT(NY5-1)
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      NC1=2
      NC2=NY5-1
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL AINT (DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - REGION 2L
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
      NC1=NTY4+1
      NC2=NTY1L-1
      CALL AINT (DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
```

```
C       - REGION 2C
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      NC1=NTY1L+1
      NC2=NTYWAL-1
      CALL AINT (DY,DZ,2,A0,A1,A2,A3,A4,CO)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C       - REGION 2R
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL AINT (DY,DZ,2,A0,A1,A2,A3,A4,CO)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C       - REGION 9
C
      DY=(Y8-Y7)/FLOAT(NY8)
      NC1=NTY7+1
      NC2=NCOL-1
      CALL AINT (DY,DZ,2,A0,A1,A2,A3,A4,CO)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C       - REGION 3
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      DZ=(ZWALL-ZFRP)/FLOAT(NZWALL)
      NC1=NTY1L+1
      NC2=NTYWAL-1
      NR1=NTZFRP+1
      NR2=NTZWAL-1
      CALL AINT (DY,DZ,5,A0,A1,A2,A3,A4,CO)
      CALL WRTMAT (A,C,NN,NR1,NR2,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C     2. CONVECTION BOUNDARIES
C
C       - REGION 5B
C
      DY=Y5/FLOAT(NY5-1)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      I=1
      NC1=2
      NC2=NY5-1
      CALL ACONV (3,1,DY,DZ,1,A0,A1,A2,A3,A4,CO)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C-  LINE 4B
C
      DY=(Y4-Y5)/FLOAT(NY4)
      NC1=NY5+1
      NC2=NTY4-1
      CALL ACONV (3,1,DY,DZ,3,A0,A1,A2,A3,A4,CO)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C       - LINE 1LB
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
      NC1=NTY4+1
      NC2=NTY1L-1
      CALL ACONV (3,1,DY,DZ,1,A0,A1,A2,A3,A4,CO)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,CO)
C
C     - LINE 1CB
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      NC1=NTY1L+1
      NC2=NTYWAL-1
      CALL ACONV (3,1,DY,DZ,1,A0,A1,A2,A3,A4,CO)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,CO)
```

```
C
C     - LINE 1RB
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL ACONV (3,1,DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - LINE 7B
C
      DY=(Y7-Y1R)/FLOAT(NY7)
      DZ=ZPOLY/(NZPOLY-1)
      NC1=NTY1R+1
      NC2=NTY7-1
      CALL ACONV (3,1,DY,DZ,4,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - LINE 8B
C
      DY=(Y8-Y7)/FLOAT(NY8)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NC1=NTY7+1
      NC2=NCOL-1
      CALL ACONV (3,1,DY,DZ,1,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - LINE 5L
C
      DY=Y5/FLOAT(NY5-1)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      J=1
      NR1=2
      NR2=NZPOLY-1
      CALL ACONV (1,8,DZ,DY,1,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C       - LINE 6L
C
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL ACONV (1,8,DZ,DY,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C       - LINE 6T
C
      DY=Y5/FLOAT(NY5-1)
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      I=NTZFRP
      NC1=2
      NC2=NY5-1
      CALL ACONV (4,7,DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - LINE 4T
C
      DY=(Y4-Y5)/FLOAT(NY4)
      NC1=NY5+1
      NC2=NTY4-1
      CALL ACONV (4,7,DY,DZ,3,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
```

```
C
C        - LINE 2LT
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
      NC1=NTY4+1
      NC2=NTY1L-1
      CALL ACONV (4,7,DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - LINE 3L
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      DZ=(ZWALL-ZFRP)/FLOAT(NZWALL)
      NR1=NTZFRP+1
      NR2=NTZWAL-1
      J=NTY1L
      CALL ACONV (1,6,DZ,DY,5,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C        - LINE 3T
C
      I=NTZWAL
      NC1=NTY1L+1
      NC2=NTYWAL-1
      CALL ACONV (4,5,DY,DZ,5,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - LINE 3R
C
      NR1=NTZFRP+1
      NR2=NTZWAL-1
      J=NTYWAL
      CALL ACONV (2,4,DZ,DY,5,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C        - LINE 2RT
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      I=NTZFRP
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL ACONV (4,3,DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - LINE 7T
C
      DY=(Y7-Y1R)/FLOAT(NY7)
      NC1=NTY1R+1
      NC2=NTY7-1
      CALL ACONV (4,3,DY,DZ,4,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - LINE 9T
C
      DY=(Y8-Y7)/FLOAT(NY8)
      NC1=NTY7+1
      NC2=NCOL-1
      CALL ACONV (4,3,DY,DZ,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - LINE 9R
C
      J=NCOL
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL ACONV (2,2,DZ,DY,2,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
```

```
C        - LINE 8R
C
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NR1=2
      NR2=NZPOLY-1
      CALL ACONV (2,2,DZ,DY,1,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C
C   INTERFACE CONDITIONS
C
C      LINE  5-6
C
      DX=(Y5)/FLOAT(NY5-1)
      DXA=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      DXB=ZPOLY/FLOAT(NZPOLY-1)
      I=NZPOLY
      NC1=2
      NC2=NY5-1
      MA=2
      MB=1
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C     INTERFACE 4L4U
C
      DX=(Y4-Y5)/NY4
      NC1=NY5+1
      NC2=NTY4-1
      MA=3
      MB=3
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 12L
C
      NC1=NTY4+1
      NC2=NTY1L-1
      MA=2
      MB=1
      DX=(Y1L-Y4)/FLOAT(NY1L)
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C  - INTERFACE 12C
C
      NC1=NTY1L+1
      NC2=NTYWAL-1
      DX=(YWALL-Y1L)/FLOAT(NYWALL)
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 12R
C
      NC1=NTYWAL+1
      NC2=NTY1R-1
      DX=(Y1R-YWALL)/FLOAT(NY1R)
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 7L7U
C
      DX=(Y7-Y1R)/NY7
      NC1=NTY1R+1
      NC2=NTY7-1
      MA=4
      MB=4
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
```

```
C        - INTERFACE 89
C
      NC1=NTY7+1
      NC2=NCOL-1
      DX=(Y8-Y7)/FLOAT(NY8)
      MA=2
      MB=1
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - 23
C
      I=NTZFRP
      NC1=NTY1L+1
      NC2=NTYWAL-1
      MA=5
      MB=2
      DX=(YWALL-Y1L)/FLOAT(NYWALL)
      DXA=(ZWALL-ZFRP)/FLOAT(NZWALL)
      DXB=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      CALL AINF (1,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT1 (A,C,NN,I,NC1,NC2,A0,A1,A2,A3,A4,C0)
C
C        - INTERFACE 54
C
      J=NY5
      NR1=2
      NR2=NZPOLY-1
      DX=ZPOLY/FLOAT(NZPOLY-1)
      DXA=Y5/FLOAT(NY5-1)
      DXB=(Y4-Y5)/FLOAT(NY4)
      MA=1
      MB=3
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C        - INTERFACE 64
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      DX=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      MA=2
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C        - INTERFACE 41L
C
      NR1=2
      NR2=NZPOLY-1
      J=NTY4
      DX=ZPOLY/(NZPOLY-1)
      DXA=(Y4-Y5)/NY4
      DXB=(Y1L-Y4)/FLOAT(NY1L)
      MA=3
      MB=1
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C        - INTERFACE 42L
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      DX=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      MA=3
      MB=2
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
```

```
C       - INTERFACE 1L1C
C
      NR1=2
      NR2=NZPOLY-1
      J=NTY1L
      DX=(ZPOLY)/FLOAT(NZPOLY-1)
      DXA=(Y1L-Y4)/NY1L
      DXB=(YWALL-Y1L)/NYWALL
      MA=1
      MB=1
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 2L2C
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      DX=(ZFRP-ZPOLY)/NZFRP
      MA=2
      MB=2
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 1C1R
C
      NR1=2
      NR2=NZPOLY-1
      J=NTYWAL
      DX=(ZPOLY)/(NZPOLY-1)
      DXA=(YWALL-Y1L)/NYWALL
      DXB=(Y1R-YWALL)/NY1R
      MA=1
      MB=1
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 2C2R
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      J=NTYWAL
      DX=(ZFRP-ZPOLY)/NZFRP
      MA=2
      MB=2
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 1R7
C
      NR1=2
      NR2=NZPOLY-1
      J=NTY1R
      DX=ZPOLY/(NZPOLY-1)
      DXA=(Y1R-YWALL)/NY1R
      DXB=(Y7-Y1R)/NY7
      MA=1
      MB=4
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 2R7
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      DX=(ZFRP-ZPOLY)/NZFRP
      MA=2
      MB=4
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J.A0,A1,A2,A3,A4,C0)
C
```

```
C       - INTERFACE 78
C
      NR1=2
      NR2=NZPOLY-1
      J=NTY7
      DX=ZPOLY/(NZPOLY-1)
      DXA=(Y7-Y1R)/NY7
      DXB=(Y8-Y7)/NY8
      MA=4
      MB=1
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C       - INTERFACE 79
C
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      DX=(ZFRP-ZPOLY)/NZFRP
      MB=2
      CALL AINF (3,MA,MB,DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WMAT2 (A,C,NN,NR1,NR2,J,A0,A1,A2,A3,A4,C0)
C
C     4. CORNERS
C
C       - INTERIOR CORNERS, ALL CONDUCTION
C
C           - 654L4U
C
      I=NZPOLY
      J=NY5
      DX1=Y5/(NY5-1)
      DX2=(Y4-Y5)/NY4
      DX3=ZPOLY/(NZPOLY-1)
      DX4=(ZFRP-ZPOLY)/NZFRP
      MA=3
      MB=2
      MC=1
      MD=3
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
C           - 4L4U1L2L
C
      I=NZPOLY
      J=NTY4
      DX1=(Y4-Y5)/NY4
      DX2=(Y1L-Y4)/NY1L
      MA=2
      MB=3
      MC=3
      MD=1
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
C           - 2C2L1L1C
C
      J=NTY1L
      DX1=(Y1L-Y4)/NY1L
      DX2=(YWALL-Y1L)/NYWALL
      MA=2
      MB=2
      MC=1
      MD=1
C
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
```

```
C         - 2R2C1C1R
C
      J=NTYWAL
      DX1=(YWALL-Y1L)/NYWALL
      DX2=(Y1R-YWALL)/NY1R
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
C     - 72R1R
C
      J=NTY1R
      DX1=(Y1R-YWALL)/NY1R
      DX2=(Y7-Y1R)/NY7
      MA=4
      MB=2
      MC=1
      MD=4
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
C     - 978
C
      J=NTY7
      DX1=(Y7-Y1R)/NY7
      DX2=(Y8-Y7)/NY8
      MA=2
      MB=4
      MC=4
      MD=1
      CALL AIC1 (MA,MB,MC,MD,DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
C  - REENTRANT CORNERS
C
      I=NTZFRP
      J=NTY1L
      N1=1
      N2=4
      NH1=7
      NH2=6
      DX1=(Y1L-Y4)/NY1L
      DX2=(YWALL-Y1L)/NYWALL
      DX3=(ZFRP-ZPOLY)/NZFRP
      DX4=(ZWALL-ZFRP)/NZWALL
      MA=5
      MB=6
      MC=2
      MD=2
      CALL AIC2 (N1,N2,NH1,NH2,MA,MB,MC,MD,
     * DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      J=NTYWAL
      N1=2
      N2=4
      NH1=3
      NH2=4
      DX1=(YWALL-Y1L)/NYWALL
      DX2=(Y1R-YWALL)/NY1R
      MA=6
      MB=5
      MC=2
      MD=2
      CALL AIC2 (N1,N2,NH1,NH2,MA,MB,MC,MD,
     * DX1,DX2,DX3,DX4,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
```

```
C       - CONVECTION ON ONE SIDE
C
      J=1
      I=NZPOLY
      N=1
      NH=8
      DX=Y5/(NY5-1)
      DXA=ZPOLY/(NZPOLY-1)
      DXB=(ZFRP-ZPOLY)/NZFRP
      MA=1
      MB=2
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=1
      J=NY5
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=Y5/(NY5-1)
      DXA=(Y4-Y5)/NY4
      MB=1
      MA=3
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      N=4
      NH=7
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y4-Y5)/NY4
      MA=2
      MB=3
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=1
      J=NTY4
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y4-Y5)/NY4
      DXA=(Y1L-Y4)/NY1L
      MB=3
      MA=1
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      N=4
      NH=7
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y1L-Y4)/NY1L
      MA=3
      MB=2
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
```

```
C
      I=1
      J=NTY1L
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y1L-Y4)/NY1L
      DXA=(YWALL-Y1L)/NYWALL
      MA=1
      MB=1
      CALL AIC3 (N,NH,MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      J=NTYWAL
      DXB=(YWALL-Y1L)/NYWALL
      DXA=(Y1R-YWALL)/NY1R
      CALL AIC3 (N,NH,MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      J=NTY1R
      DXB=(Y1R-YWALL)/NY1R
      DXA=(Y7-Y1R)/NY7
      MB=1
      MA=4
      CALL AIC3 (N,NH.MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      N=4
      NH=3
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y7-Y1R)/NY7
      MA=2
      MB=4
      CALL AIC3 (N,NH,MA,MB,
     * DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=1
      J=NTY7
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y7-Y1R)/NY7
      DXA=(Y8-Y7)/NY8
      MB=4
      MA=1
      CALL AIC3 (N,NH,MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      N=4
      NH=3
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y8-Y7)/NY8
      MA=4
      MB=2
      CALL AIC3 (N,NH,MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NZPOLY
      J=NCOL
      N=2
      NH=2
      DX=(Y8-Y7)/NY8
      DXB=ZPOLY/(NZPOLY-1)
      DXA=(ZFRP-ZPOLY)/NZFRP
      MB=1
      MA=2
      CALL AIC3 (N,NH,MA,MB,
     *  DX,DXA,DXB,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
```

```
C  - EXTERIOR CORNERS
C
      I=1
      J=1
      N1=1
      N2=3
      NH1=8
      NH2=1
      DX1=Y5/(NY5-1)
      DX2=ZPOLY/(NZPOLY-1)
      M=1
      CALL AIC4 (N1,N2,NH1,NH2,M,
     *  DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=1
      J=NCOL
      N1=2
      N2=3
      NH1=2
      NH2=1
      DX1=(Y8-Y7)/NY8
      CALL AIC4 (N1,N2,NH1,NH2,M,
     *  DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      N1=2
      N2=4
      NH1=2
      NH2=3
      DX2=(ZFRP-ZPOLY)/NZFRP
      M=2
      CALL AIC4 (N1,N2,NH1,NH2,M,
     *  DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZWAL
      J=NTYWAL
      N1=2
      N2=4
      NH1=4
      NH2=5
      DX1=(YWALL-Y1L)/NYWALL
      DX2=(ZWALL-ZFRP)/NZWALL
      M=5
      CALL AIC4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      J=NTY1L
      N1=1
      N2=4
      NH1=6
      NH2=5
      CALL AIC4 (N1,N2,NH1,NH2,M,
     *  DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      I=NTZFRP
      J=1
      N1=1
      N2=4
      NH1=8
      NH2=7
      DX1=Y5/(NY5-1)
      DX2=(ZFRP-ZPOLY)/NZFRP
      M=2
      CALL AIC4 (N1,N2,NH1,NH2,M,
     *  DX1,DX2,A0,A1,A2,A3,A4,C0)
      CALL WPT (A,C,NN,I,J,A0,A1,A2,A3,A4,C0)
C
      RETURN
C
      END
```

```
C==============================================================================
C
      SUBROUTINE BCONV (N,M,DY,DZ,MM,B0)
C
C******************************************************************************
C       *     SUBROUTINE TO FORM B VECTOR - LINES                            *
C******************************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      B0=HTC(M)*DY*(ALFA*TAMBP(M)+(1.-ALFA)*TAMB(M))
C
      RETURN
C
      END


C
C******************************************************************************
C
      SUBROUTINE B2 (N1,N2,NH1,NH2,MA,MB,MC,MD,DX1,DX2,DX3,DX4,B0)
C
C******************************************************************************
C  *     SUBROUTINE TO FORM B VECTOR - REENTRANT CORNERS                      *
C******************************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      DDX1=DX1
      DDX2=DX4
      IF (N1.EQ.2) DDX1=DX2
      A0=0.5*HTC(NH1)*DDX1
      A01=0.5*HTC(NH2)*DDX2
      B0=A0*(ALFA*TAMBP(NH1)+(1.-ALFA)*TAMB(NH1)) +
     *   A01*(ALFA*TAMBP(NH2)+(1.-ALFA)*TAMB(NH2))
C
      RETURN
C
      END

C
C******************************************************************************
C
      SUBROUTINE B3 (N,NH,MA,MB,DX,DXA,DXB,B0)
C
C******************************************************************************
C*     SUBROUTINE TO FORM B VECTOR - CORNER NODE, CONVECTION 1 SIDE
C******************************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      B0=0.5*HTC(NH)*(DXA+DXB)*(ALFA*TAMBP(NH)+(1.-ALFA)*TAMB(NH))
C
      RETURN
C
      END

C
C******************************************************************************
C
      SUBROUTINE B4 (N1,N2,NH1,NH2,M,DX1,DX2,B0)
C
C******************************************************************************
C*     EXTERIOR CORNERS WITH HEAT TRANSFER - B VECTOR                         *
C******************************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      A0=0.5*DX2*HTC(NH1)
      A01=0.5*DX1*HTC(NH2)
      B0=A0*(ALFA*TAMBP(NH1)+(1.-ALFA)*TAMB(NH1)) +
     *   A01*(ALFA*TAMBP(NH2)+(1.-ALFA)*TAMB(NH2))
C
      RETURN
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE WB2 (B,NN,NR1,NR2,J,BO)
C
C***********************************************************************
C*     SUBROUTINE TO WRITE A ROW OF NODES INTO THE B VECTOR           *
C***********************************************************************
C
      DIMENSION B(NN),N(5)
C
      DO 100 I=NR1,NR2
      CALL GETNOD (I,J,N)
  100 B(N(1))=B(N(1))+BO
      RETURN
C
      END

C
C***********************************************************************
C
      SUBROUTINE WB1 (B,NN,I,NC1,NC2,BO)
C
C***********************************************************************
C*     SUBROUTINE TO WRITE A COLUMN OF NODES IN THE B VECTOR          *
C***********************************************************************
C
      DIMENSION B(NN),N(5)
C
      DO 100 J=NC1,NC2
      CALL GETNOD (I,J,N)
  100 B(N(1))=B(N(1))+BO
      RETURN
C
      END


C
C*****************************************************************
C
      SUBROUTINE WB (B,NN.I.J.BO)
C
C*****************************************************************
C*       SUBROUTINE TO WRITE A POINT IN THE B VECTOR             *
C*****************************************************************
C
      DIMENSION B(NN).N(5)
C
      CALL GETNOD(I,J.N)
      B(N(1))=B(N(1))+BO
      RETURN
C
      END
```

```
C
C************************************************************************
C
      SUBROUTINE FORMB (ICON,B,BTEMP,NN)
C
C************************************************************************
C*       SUBROUTINE TO FORM CONVECTION PART OF RHS VECTOR              *
C************************************************************************
C
      COMMON /NUMZ/ NZPOLY,NZFRP,NZWALL
      COMMON /NUMY/ NY5,NY4,NY1L,NYWALL,NY1R,NY7,NY8
      COMMON /GRIDBY/ NTY4,NTY1L,NTYWAL,NTY1R,NTY7,NCOL
      COMMON /GRIDBZ/ NTZFRP,NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP,ZWALL
      COMMON /YCOOR/ Y5,Y4,Y1L,YWALL,Y1R,Y7,Y8
C
      DIMENSION B(NN),BTEMP(NN)
C
C    ICON=-1  -  SET B=0, REFORM RHS INTO BTEMP, REWRITE BTEMP INTO B
C    ICON= 0  -  FORM RHS INTO BTEMP, WRITE BTEMP INTO B
C    ICON= 1  -  SET B=0, WRITE BTEMP INTO B
C
      IF (ICON.EQ.0) GO TO 100
C
C      ZERO B VECTOR
C
      DO 10 I=1,NN
   10 B(I)=0.
      IF (ICON.EQ.1) GO TO 200
C
C      FORM CONVECTION LOAD VECTOR BTEMP
C
C      2. CONVECTION BOUNDARIES
C
C        - REGION 5B
C
  100 DY=Y5/FLOAT(NY5-1)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      I=1
      NC1=2
      NC2=NY5-1
      CALL BCONV (3,1,DY,DZ,1,B0)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,B0)
C
C-  LINE 4B
C
      DY=(Y4-Y5)/FLOAT(NY4)
      NC1=NY5+1
      NC2=NTY4-1
      CALL BCONV (3,1,DY,DZ,3,B0)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,B0)
C
C        - LINE 1LB
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
      NC1=NTY4+1
      NC2=NTY1L-1
      CALL BCONV (3,1,DY,DZ,1,B0)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,B0)
C
C      - LINE 1CB
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      NC1=NTY1L+1
      NC2=NTYWAL-1
```

```
      CALL BCONV (3,1,DY,DZ,1,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C      - LINE 1RB
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL BCONV (3,1,DY,DZ,1,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 7B
C
      DY=(Y7-Y1R)/FLOAT(NY7)
      NC1=NTY1R+1
      NC2=NTY7-1
      CALL BCONV (3,1,DY,DZ,4,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 8B
C
      DY=(Y8-Y7)/FLOAT(NY8)
      NC1=NTY7+1
      NC2=NCOL-1
      CALL BCONV (3,1,DY,DZ,1,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 5L
C
      DY=Y5/FLOAT(NY5-1)
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      J=1
      NR1=2
      NR2=NZPOLY-1
      CALL BCONV (1,8,DZ,DY,1,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C        - LINE 6L
C
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL BCONV (1,8,DZ,DY,2,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C        - LINE 6T
C
      DY=Y5/FLOAT(NY5-1)
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      I=NTZFRP
      NC1=2
      NC2=NY5-1
      CALL BCONV (4,7,DY,DZ,2,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 4T
C
      DY=(Y4-Y5)/FLOAT(NY4)
      NC1=NY5+1
      NC2=NTY4-1
      CALL BCONV (4,7,DY,DZ,3,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 2LT
C
      DY=(Y1L-Y4)/FLOAT(NY1L)
```

```
      NC1=NTY4+1
      NC2=NTY1L-1
      CALL BCONV (4,7,DY,DZ,2,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 3L
C
      DY=(YWALL-Y1L)/FLOAT(NYWALL)
      DZ=(ZWALL-ZFRP)/FLOAT(NZWALL)
      NR1=NTZFRP+1
      NR2=NTZWAL-1
      J=NTY1L
      CALL BCONV (1,6,DZ,DY,5,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C        - LINE 3T
C
      I=NTZWAL
      NC1=NTY1L+1
      NC2=NTYWAL-1
      CALL BCONV (4,5,DY,DZ,5,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 3R
C
      NR1=NTZFRP+1
      NR2=NTZWAL-1
      J=NTYWAL
      CALL BCONV (2,4,DZ,DY,5,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C        - LINE 2RT
C
      DY=(Y1R-YWALL)/FLOAT(NY1R)
      DZ=(ZFRP-ZPOLY)/FLOAT(NZFRP)
      I=NTZFRP
      NC1=NTYWAL+1
      NC2=NTY1R-1
      CALL BCONV (4,3,DY,DZ,2,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 7T
C
      DY=(Y7-Y1R)/FLOAT(NY7)
      NC1=NTY1R+1
      NC2=NTY7-1
      CALL BCONV (4,3,DY,DZ,4,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 9T
C
      DY=(Y8-Y7)/FLOAT(NY8)
      NC1=NTY7+1
      NC2=NCOL-1
      CALL BCONV (4,3,DY,DZ,2,BO)
      CALL WB1 (BTEMP,NN,I,NC1,NC2,BO)
C
C        - LINE 9R
C
      J=NCOL
      NR1=NZPOLY+1
      NR2=NTZFRP-1
      CALL BCONV (2,2,DZ,DY,2,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C        - LINE 8R
```

```
C
      DZ=ZPOLY/FLOAT(NZPOLY-1)
      NR1=2
      NR2=NZPOLY-1
      CALL BCONV (2,2,DZ,DY,1,BO)
      CALL WB2 (BTEMP,NN,NR1,NR2,J,BO)
C
C
C  - REENTRANT CORNERS
C
      I=NTZFRP
      J=NTY1L
      N1=1
      N2=4
      NH1=7
      NH2=6
      DX1=(Y1L-Y4)/NY1L
      DX2=(YWALL-Y1L)/NYWALL
      DX3=(ZFRP-ZPOLY)/NZFRP
      DX4=(ZWALL-ZFRP)/NZWALL
      MA=5
      MB=6
      MC=2
      MD=2
      CALL B2 (N1,N2,NH1,NH2,MA,MB,MC,MD,
     *  DX1,DX2,DX3,DX4,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      J=NTYWAL
      N1=2
      N2=4
      NH1=3
      NH2=4
      DX1=(YWALL-Y1L)/NYWALL
      DX2=(Y1R-YWALL)/NY1R
      MA=6
      MB=5
      MC=2
      MD=2
      CALL B2 (N1,N2,NH1,NH2,MA,MB,MC,MD,
     *  DX1,DX2,DX3,DX4,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
C        - CONVECTION ON ONE SIDE
C
      J=1
      I=NZPOLY
      N=1
      NH=8
      DX=Y5/(NY5-1)
      DXA=ZPOLY/(NZPOLY-1)
      DXB=(ZFRP-ZPOLY)/NZFRP
      MA=1
      MB=2
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=1
      J=NY5
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=Y5/(NY5-1)
      DXA=(Y4-Y5)/NY4
      MB=1
```

```
      MA=3
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=NTZFRP
      N=4
      NH=7
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y4-Y5)/NY4
      MA=2
      MB=3
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=1
      J=NTY4
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y4-Y5)/NY4
      DXA=(Y1L-Y4)/NY1L
      MB=3
      MA=1
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=NTZFRP
      N=4
      NH=7
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y1L-Y4)/NY1L
      MA=3
      MB=2
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J.BO)
C
      I=1
      J=NTY1L
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y1L-Y4)/NY1L
      DXA=(YWALL-Y1L)/NYWALL
      MA=1
      MB=1
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      J=NTYWAL
      DXB=(YWALL-Y1L)/NYWALL
      DXA=(Y1R-YWALL)/NY1R
      CALL B3 (N,NH,MA,MB,
     * DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      J=NTY1R
      DXB=(Y1R-YWALL)/NY1R
      DXA=(Y7-Y1R)/NY7
      MB=1
```

```
      MA=4
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J.BO)
C
      I=NTZFRP
      N=4
      NH=3
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y7-Y1R)/NY7
      MA=2
      MB=4
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J.BO)
C
      I=1
      J=NTY7
      N=3
      NH=1
      DX=ZPOLY/(NZPOLY-1)
      DXB=(Y7-Y1R)/NY7
      DXA=(Y8-Y7)/NY8
      MB=4
      MA=1
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=NTZFRP
      N=4
      NH=3
      DX=(ZFRP-ZPOLY)/NZFRP
      DXA=DXB
      DXB=(Y8-Y7)/NY8
      MA=4
      MB=2
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J,BO)
C
      I=NZPOLY
      J=NCOL
      N=2
      NH=2
      DX=(Y8-Y7)/NY8
      DXB=ZPOLY/(NZPOLY-1)
      DXA=(ZFRP-ZPOLY)/NZFRP
      MB=1
      MA=2
      CALL B3 (N,NH,MA,MB,
     *  DX,DXA,DXB,BO)
      CALL WB (BTEMP,NN,I,J.BO)
C
C  - EXTERIOR CORNERS
C
      I=1
      J=1
      N1=1
      N2=3
      NH1=8
      NH2=1
      DX1=Y5/(NY5-1)
      DX2=ZPOLY/(NZPOLY-1)
      M=1
```

```
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J.B0)
C
      I=1
      J=NCOL
      N1=2
      N2=3
      NH1=2
      NH2=1
      DX1=(Y8-Y7)/NY8
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J.B0)
C
      I=NTZFRP
      N1=2
      N2=4
      NH1=2
      NH2=3
      DX2=(ZFRP-ZPOLY)/NZFRP
      M=2
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J.B0)
C
      I=NTZWAL
      J=NTYWAL
      N1=2
      N2=4
      NH1=4
      NH2=5
      DX1=(YWALL-Y1L)/NYWALL
      DX2=(ZWALL-ZFRP)/NZWALL
      M=5
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J,B0)
C
      J=NTY1L
      N1=1
      N2=4
      NH1=6
      NH2=5
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J.B0)
C
      I=NTZFRP
      J=1
      N1=1
      N2=4
      NH1=8
      NH2=7
      DX1=Y5/(NY5-1)
      DX2=(ZFRP-ZPOLY)/NZFRP
      M=2
      CALL B4 (N1,N2,NH1,NH2,M,
     * DX1,DX2,B0)
      CALL WB (BTEMP,NN,I,J,B0)
C
  200 CONTINUE
C
C      WRITE BTEMP INTO B VECTOR
C
      DO 300 I=1,NN
  300 B(I)=BTEMP(I)
      RETURN
C
      END
```

```
C
C*****************************************************************
C
      SUBROUTINE QSOLV2 (ISTEP.KOUNT.T.NN.TIME)
C
C********************************************************************
C*     SUBROUTINE TO EFFECT ENERGY BALANCES                         *
C********************************************************************
C
      COMMON /NUMZ/ NZPOLY,NZFRP.NZWALL
      COMMON /NUMY/ NY5.NY4.NY1L.NYWALL.NY1R.NY7.NY8
      COMMON /GRIDBY/ NTY4.NTY1L.NTYWAL.NTY1R.NTY7.NCOL
      COMMON /GRIDBZ/ NTZFRP.NTZWAL
      COMMON /ZCOOR/ ZPOLY,ZFRP.ZWALL
      COMMON /YCOOR/ Y5.Y4.Y1L.YWALL.Y1R.Y7.Y8
      COMMON /MATS/ RKAY(6).RHO(6).CP(6)
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /HEAT/ Q(21),IEB,IQPR,ICHECK(5)
C
      DIMENSION T(NN),HT(13).N(5)
C
      CALL HEADER1
      WRITE (6.3001) TIME
C
      HT(1)=Q(1)+Q(2)+Q(3)
      HT(2)=Q(4)
      HT(3)=Q(5)+Q(6)+Q(7)
      HT(4)=Q(8)+Q(9)+Q(10)
      HT(5)=Q(11)
      HT(6)=Q(12)
      HT(7)=Q(13)+Q(14)+Q(15)
      HT(8)=Q(16)
      HT(9)=Q(17)
      HT(10)=Q(18)
      HT(11)=Q(19)
      HT(12)=Q(20)
      HT(13)=Q(21)
      IF (ICHECK(1).EQ.0) GO TO 100
      WRITE (6.2003) (HT(I),I=1.13)
C
C     HEAT BALANCES FOR VARIOUS REGIONS
C
C
C     REGION 5-6-4L-4U-1L-2L
C
      H1=HT(7)-HT(8)+HT(1)-HT(12)
C
C     REGION 1C
C
  100 IF (ICHECK(2).EQ.0) GO TO 200
      H2=HT(8)-HT(11)+HT(2)-HT(9)
C
C     REGION 1R-2R-7L-7U-8-9
C
  200 IF (ICHECK(3).EQ.0) GO TO 300
      H3=HT(9)+HT(10)+HT(4)+HT(3)
C
C     REGION 2C
C
  300 IF (ICHECK(4).EQ.0) GO TO 400
      H4=-HT(13)+HT(12)+HT(11)-HT(10)
C
C     REGION 3
C
  400 IF (ICHECK(5).EQ.0) GO TO 500
      H5=HT(6)+HT(5)+HT(13)
C
C     TEST FOR PRINTING
C
  500 IF (KOUNT.LT.IQPR) RETURN
      KOUNT=0
C
C     PRINT ENERGY BALANCES
C
      IF (ICHECK(1).GE.0) GO TO 600
      CALL HEADER1
      WRITE (6,2000) H1.H2.H3.H4.H5
C
C
```

```
C     COMPUTE FRAME COEFFICIENT
C
C        - AVERAGE MC SPECIMEN SURFACE TEMP
C
      DELTAZ=ZWALL-ZFRP
      PER=15.236
      AW=13.93
  600 QT=PER*HT(7)+AW*HT(6)/DELTAZ
C
C     PRINT NET ENERGY INTO OR OUT OF MC
C
      WRITE (6,2001) QT
C
      AHTMC=AW*HT(6)/DELTAZ
      FCMCA=1.-ABS(AHTMC)/ABS(QT)
C
C     FRAME COEFFICIENT USING 1 - D HEAT FLOW THROUGH WALL
C
      J=NTYWAL
      I=NTZWAL
      CALL GETNOD (I,J,N)
      TR=T(N(1))
      J=NTY1L
      CALL GETNOD (I,J,N)
      TL=T(N(1))
      HTW=AW*(TR-TL)*RKAY(5)/(YWALL-Y1L)
      FCHXS=1.-ABS(HTW)/ ABS(QT)
C
C
C     PRINT RELATIVE FRAME COEFICIENTS
C
      WRITE (6,2002) HTW,FCMCA,FCHXS
C
      CALL HEADER1
C
      RETURN
C
 2000 FORMAT (//
     * / 20X,"E N E R G Y   B A L A N C E S"//
     * / 15X,"H1 (REGION 5-6-4U-4L-1L-2L)     =",E14.7/
     * / 15X,"H2 (REGION 1C                )     =",E14.7/
     * / 15X,"H3 (REGION 1R-2R-7L-7U-8-9)     =",E14.7/
     * / 15X,"H4 (REGION 2C                )     =",E14.7/
     * / 15X,"H5 (REGION 3                 )     =",E14.7///)
 2001 FORMAT (//
     * / 20X,"NET MC ENERGY                  =",E14.7///)
 2002 FORMAT (//
     * / 20X,"FRAME COEFFICIENTS-REL"/
     * / 15X,"HTW                            =",E14.7/
     * / 15X,"FCMCA                          =",E14.7/
     * / 15X,"FCHXS                          =",E14.7///)
 2003 FORMAT (//
     * / 20X,"T O T A L   L I N E   Q ' S"//
     * / 15X,"HT (1 )                        =",E14.7/
     * / 15X,"HT (2 )                        =",E14.7/
     * / 15X,"HT (3 )                        =",E14.7/
     * / 15X,"HT (4 )                        =",E14.7/
     * / 15X,"HT (5 )                        =",E14.7/
     * / 15X,"HT (6 )                        =",E14.7/
     * / 15X,"HT (7 )                        =",E14.7/
     * / 15X,"HT (8 )                        =",E14.7/
     * / 15X,"HT (9 )                        =",E14.7/
     * / 15X,"HT (10)                        =",E14.7/
     * / 15X,"HT (11)                        =",E14.7/
     * / 15X,"HT (12)                        =",E14.7/
     * / 15X,"HT (13)                        =",E14.7///)
 3001 FORMAT (//
     * / 20X,"E N E R G Y   B A L A N C E S   A T"/
     * / 20X,"          T I M E   = ",E14.7///)
C
      END
```

```
C***********************************************************************
C
      SUBROUTINE QCONVH (NH,I,NC1,NC2,DX,T,NN,F)
C
C***********************************************************************
C*     SUBROUTINE TO COMPUTE HORIZONTAL CONVECTIVE Q'S                 *
C***********************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      DIMENSION T(NN),N(5)
C
      TI=ALFA*TAMBP(NH)+(1.-ALFA)*TAMB(NH)
      NDIV=NC2-NC1
      IF (NDIV.GE.1) GO TO 100
      CALL HEADER1
      WRITE (6,3000) I,NC1,NC2,NDIV
      CALL ADIOS (2)
C
  100 J1=NC1+1
      J2=NC2-1
      SUM=0.
C
C     FIRST NODE
C
      J=NC1
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(TI-T(N(1)))
C
C     LAST NODE
C
      J=NC2
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(TI-T(N(1)))
C
C     INTERIOR NODES
C
      DO 200 J=J1,J2
      CALL GETNOD (I,J,N)
  200 SUM=SUM+(TI-T(N(1)))
C
      F=HTC(NH)*DX*SUM
C
      RETURN
C
 3000 FORMAT (//15X,"--- ERROR TERMINATION ---"/
     * / 20X,"FROM QCONVH - NDIV=0"/
     * / 20X,"I,NC1,NC2,NDIV = ",5I5)
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE QCONVV (NH,NR1,NR2,J,DX,T,NN,F)
C
C*********************************************************************
C*     SUBROUTINE TO COMPUTE HORIZONTAL CONVECTIVE Q'S               *
C*********************************************************************
C
      COMMON /HTCOEF/ HTC(8),TAMB(8),TAMBP(8)
      COMMON /TIM/ TIMO,TIMF,DELTA,ALFA,IRFINE
C
      DIMENSION T(NN),N(5)
C
      TI=ALFA*TAMBP(NH)+(1.-ALFA)*TAMB(NH)
      NDIV=NR2-NR1
      IF (NDIV.GE.1) GO TO 100
      CALL HEADER1
      WRITE (6,3000) NR1,NR2,J,NDIV
      CALL ADIOS (2)
C
  100 I1=NR1+1
      I2=NR2-1
      SUM=0.
C
C     FIRST NODE
C
      I=NR1
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(TI-T(N(1)))
C
C     LAST NODE
C
      I=NR2
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(TI-T(N(1)))
C
C     INTERIOR NODES
C
      DO 200 I=I1,I2
      CALL GETNOD (I,J,N)
  200 SUM=SUM+(TI-T(N(1)))
C
      F=HTC(NH)*DX*SUM
C
      RETURN
C
 3000 FORMAT (//15X,"--- ERROR TERMINATION ---"/
     * / 20X,"FROM QCONVV - NDIV=0"/
     * / 20X,"NR1,NR2,J,NDIV = ",5I5)
C
      END
```

```
C
C*********************************************************************
C
      SUBROUTINE QCONDV (M,NR1,NR2,J,ND,DX,DY,C,T,NN,F)
C
C*********************************************************************
C*     SUBROUTINE TO COMPUTE VERTICAL CONDUCTION FLUX                 *
C*********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION T(NN),N(5)
C
      NDIV=NR2-NR1
      IF (NDIV.GE.1) GO TO 100
      CALL HEADER1
      WRITE (6,3000) NR1,NR2,J,NDIV
      CALL ADIOS (2)
  100 I1=NR1+1
      I2=NR2-1
      SUM=0.
C
C     FIRST POINT
C
      I=NR1
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(T(N(1))-T(N(1)+ND))
C
C     LAST NODE
C
      I=NR2
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(T(N(1))-T(N(1)+ND))
C
C     INTERIOR NODES
C
      DO 200 I=I1,I2
      CALL GETNOD (I,J,N)
  200 SUM=SUM+T(N(1))-T(N(1)+ND)
C
      F=C*RKAY(M)*DY*SUM/(DX)
C
C
C     WRITE DEBUG INFO
C
      IF (IDBUG(8).EQ.0) RETURN
      WRITE (6,2000) M,NR1,NR2,J,F
C
      RETURN
C
 2000 FORMAT (//
     * / 20X,"IN QCONDV - DEBUG INFO"/
     * / 15X,"M,NR1,NR2,J,F = ",4I5,5X,E14.7//)
 3000 FORMAT (//15X,"--- ERROR TERMINATION ---"/
     * / 20X,"IN QCONDV - NDIV=0"/
     * / 20X,"NR1,NR2,J,NDIV = ",5I5)
C
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE QCONDH (M,I,NC1,NC2,ND,DX,DY,C,T,NN,F)
C
C***********************************************************************
C*     SUBROUTINE TO COMPUTE HORIZONTAL CONDUCTION FLUX                *
C***********************************************************************
C
      COMMON /MATS/ RKAY(6),RHO(6),CP(6)
      COMMON /DBUG/ IDBUG(10)
C
      DIMENSION T(NN),N(5)
C
      NDIV=NC2-NC1
      IF (NDIV.GE.1) GO TO 100
      CALL HEADER1
      WRITE (6,3000) I,NC1,NC2,NDIV
      CALL ADIOS (2)
  100 J1=NC1+1
      J2=NC2-1
      SUM=0.
C
C     FIRST POINT
C
      J=NC1
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(T(N(1))-T(N(1)+ND))
C
C     LAST NODE
C
      J=NC2
      CALL GETNOD (I,J,N)
      SUM=SUM+0.5*(T(N(1))-T(N(1)+ND))
C
C     INTERIOR NODES
C
      DO 200 J=J1,J2
      CALL GETNOD (I,J,N)
  200 SUM=SUM+T(N(1))-T(N(1)+ND)
C
      F=C*RKAY(M)*DY*SUM/(DX)
C
C     WRITE DEBUG INFO
C
      IF (IDBUG(8).EQ.0) RETURN
      WRITE (6,2000) M,I,NC1,NC2,F
 2000 FORMAT (//
     * / 20X,"IN QCONDH - DEBUG INFO"/
     * / 15X,"M,I,NC1,NC2,F = ",4I5,5X,E14.7//)
C
      RETURN
C
 3000 FORMAT (//15X,"--- ERROR TERMINATION ---"/
     * / 20X,"IN QCONDH - NDIV=0"/
     * / 20X,"I,NC1,NC2,NDIV = ",5I5)
C
      END
```

```
************************************************************************

      SUBROUTINE PLOTH (IND,TIME,T,NN)

************************************************************************
*     SUBROUTINE TO PERFORM CONTOUR PLOTS INPUT                        *
************************************************************************

      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
      COMMON /PHIST/ ITITH(5,72),ITITXH(5,36),ITITYH(5,36),MH(5),
     *              NODESH(5,5),N,XMIN,XMAX

      DIMENSION T(NN)
      DIMENSION ITITLE(144),YH(81,5,5),Y(81,5),X(81),IMAG4(5151)

      DATA INC,IY /1,81/

      KH=KH+1
      X(KH)=TIME

      FILL PLOT ARRAY

      DO 100 I=1,NUMCON
      M=MH(I)
      DO 100 J=1,M
      NODE=NODESH(I,J)
  100 YH(KH,J,I)=T(NODE)

      IF (IND.EQ.0) RETURN

      PERFORM PLOTTING

      CALL HEADER1
      WRITE (6,4000) TIME

      DO 400 I=1,NUMHIS
      M=MH(I)

      FILL TITLE

      L=0
      DO 210 J=1,72
      L=L+1
  210 ITITLE(L)=ITITH(I,J)
      DO 220 J=1,36
      L=L+1
  220 ITITLE(L)=ITITXH(I,J)
      DO 230 J=1,36
      L=L+1
  230 ITITLE(L)=ITITYH(I,J)

      FILL THIS PLOT ARRAY

      DO 300 L=1,N
      DO 300 J=1,M
  300 Y(J,L)=YH(L,J,I)
```

```
      SET RANGE

      RANGE(1)=XMIN
      RANGE(2)=XMAX

      PERFORM PLOT

      IF (I.NE.1) CALL HEADER1
      WRITE (6,2000) I
      DO 395 J=1,M
  395 WRITE (6,2001) J.NODESH(I,J)
  400 CALL USPLT (X,Y,IY,N,KH,INC,ITITLE,RANGE,ICHAR,IOPT,
     *            IMAG4,IER)

      RETURN

 2000 FORMAT (//25X,"C U R V E   O U T P U T"/
     * / 15X,"CURVE NO.  ",I5//
     * / 10X,"LABEL",5X,"NODE"/)
 2001 FORMAT (
     * 5(/7X,I5,10X,I5))
 4000 FORMAT (////15X,"T I M E   H I S T O R Y   P L O T S   A T"/
     * / 30X,"T I M E  =",E12.5////)
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE PLOTIN (ITRAN)
C
C***********************************************************************
C*     SUBROUTINE TO PERFORM BASE PLOT INPUT                          *
C***********************************************************************
C
      COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
     *               ICHAR(10),NSTEPC
C
C     READ PLOT PARAMETERS
C
      READ (5,1000) NUMCON,NUMHIS,IOPT,NOUT,NSTEPC
      READ (5,1001) TMIN,TMAX
      CALL HEADER1
      WRITE (6,2000) NUMCON,NUMHIS,IOPT,NOUT,NSTEPC,TMIN,TMAX
      IF (NOUT.EQ.2.OR.NOUT.EQ.6) GO TO 100
      WRITE (6,3000)
      WRITE (6,3001) NOUT
      CALL ADIOS (2)
C
  100 CALL UGETIO (3,NIN,NOUT)
C
      IF (NUMCON.LE.5) GO TO 200
      WRITE (6,3000)
      WRITE (6,3002) NUMCON
C
  200 IF (NUMHIS.LE.5) GO TO 300
      WRITE (6,3000)
      WRITE (6,3003) NUMHIS
C
C     READ CONTOUR INFORMATION
C
  300 IF (NUMCON.NE.0) CALL PLTCIN (NUMCON)
C
C     READ TIME HISTORY INPUT
C
      IF (NUMHIS.NE.0) CALL PLTHIN (ITRAN)
C
C     SET CURVE LABELS
C
      ICHAR(1)=1H
C
C     SET RANGE
C
      RANGE(3)=TMIN
      RANGE(4)=TMAX
C
```

```
      RETURN
C
 1000 FORMAT (16I5)
 1001 FORMAT (8E10.0)
 2000 FORMAT (//
     * / 20X,"P L O T   B A S E   I N F O R M A T I O N"//
     * / 15X,"NUMCON                                    =",I5/
     * / 15X,"NUMHIS                                    =",I5/
     * / 15X,"IOPT                                      =",I5/
     * / 25X," 80 CHARACTERS - IOPT = 0"/
     * / 25X,"126 CHARACTERS - IOPT = 1"/
     * / 15X,"NOUT                                      =",I5/
     * / 25X,"FILE IMSLS.OUT - NOUT = 2"/
     * / 25X,"FILE FLANK.OUT - NOUT = 5"/
     * / 15X,"STEP INTERVAL FOR CONTOUR PLOTS           =",I5/
     * / 15X,"RANGE FOR PLOTS"/
     * / 25X,"TMIN                                      =",E14.5/
     * / 25X,"TMAX                                      =",E14.5//)
 3000 FORMAT (//"---ERROR TERMINATION--- FROM PLOTIN"//)
 3001 FORMAT (15X,"NOUT = ",I5,"   SHOULD BE 2 OR 6")
 3002 FORMAT (15X,"NUMCON = ",I5,"   SHOULD BE < OR = 5")
 3003 FORMAT (15X,"NUMHIS = ",I5,"   SHOULD BE < OR = 5")
C
      END
```

```
C***********************************************************************
C
      SUBROUTINE PLOTC (IND,TIME,T,NN)
C
C**********************************************************************
C*     SUBROUTINE TO PERFORM CONTOUR PLOTS INPUT                      *
C**********************************************************************
C
     COMMON /DBUG/ IDBUG(10)
     COMMON /PLOTS/ IPLT,NUMCON,NUMHIS,IOPT,NOUT,RANGE(4),NSTEPH,
    *               ICHAR(10),NSTEPC
     COMMON /PCONT/ ITITC(5,72),ITITXC(5,36),ITITYC(5,36),NC(5),
    *               NODESC(5,20)
C
     DIMENSION T(NN)
     DIMENSION ITITLE(144),YC(20,5,5),Y(20,5),X(20),
    *          IMAG4(5151),TIMSAV(5)
C
     DATA INC,IYC /1,5/
C
 100 KC=KC+1
     IF (IND.EQ.1.OR.KC.EQ.6) GO TO 300
     TIMSAV(KC)=TIME
C
C     FILL PLOT ARRAY
C
     DO 200 I=1,NUMCON
     N=NC(I)
     DO 200 J=1,N
     NODE=NODESC(I,J)
 200 YC(J,KC,I)=T(NODE)
C
     RETURN
C
C     PERFORM PLOTTING
C
 300 KC=KC-1
     CALL HEADER1
C
     WRITE (6,4000) KC,TIME
     DO 600 I=1,NUMCON
     IF (I.NE.1) CALL HEADER1
     WRITE (6,2000) I
     DO 305 J=1,KC
 305 WRITE (6,2001) J,TIMSAV(J)
C
C     FILL TITLE
C
     L=0
     DO 310 J=1,72
     L=L+1
 310 ITITLE(L)=ITITC(I,J)
     DO 320 J=1,36
     L=L+1
 320 ITITLE(L)=ITITXC(I,J)
     DO 330 J=1,36
     L=L+1
 330 ITITLE(L)=ITITYC(I,J)
C
```

```
C     FILL THIS PLOT ARRAY
C
      N=NC(I)
      DO 400 J=1,N
      X(J)=NODESC(I,J)
      DO 400 L=1,KC
  400 Y(J,KC)=YC(J,KC,I)
C
C     PICK OUT MAX,MIN
C
      XMIN=1.E50
      DO 500 J=1,N
  500 XMIN=AMIN1(XMIN,X(J))
C
      XMAX=-1.E50
      DO 510 J=1,N
  510 XMAX=AMAX1(XMAX,X(J))
C
      RANGE(1)=XMIN
      RANGE(2)=XMAX
      IF (IDBUG(10).EQ.0) GO TO 520
      WRITE (6,5000) I,TIME
      WRITE (6,5001) X
      WRITE (6,5002) Y
      WRITE (6,5003) IYC,N,KC,INC
      WRITE (6,5004) RANGE
      WRITE (6,5005) IOPT
  520 CONTINUE
C
C     PERFORM PLOT
C
  600 CALL USPLT (X,Y,IYC,N,KC,INC,ITITLE,RANGE,ICHAR,IOPT,
     *            IMAG4,IER)
C
      KC=0
      IF (IND.NE.1) GO TO 100
C
      RETURN
C
 2000 FORMAT (//25X,"C U R V E   O U T P U T"/
     * / 15X,"CURVE NO.  ",I5//
     * / 10X,"LABEL",5X,"TIME"/)
 2001 FORMAT (
     * 5(/7X,I5,6X,E12.5))
 4000 FORMAT (15X,I5," NO.  O F   C O N T O U R   P L O T S   A T"/
     * / 30X,"T I M E  =",E12.5////)
 5000 FORMAT (5X,"DEBUG(10) OUTPUT, PLOT NO.",I5,"  TIME =",E12.5//)
 5001 FORMAT (5X,"X - MATRIX ",/10(/10(1X,E11.4)))
 5002 FORMAT (5X,"Y - MATRIX ",/10(/10(1X,E11.4)))
 5003 FORMAT (5X,"IYC,N,KC,INC =",16I5)
 5004 FORMAT (5X,"RANGE ",5(2X,E12.4))
 5005 FORMAT (5X,"IOPT = ",I5)
      END
```

```
C
C***********************************************************************
C
      SUBROUTINE PLTCIN (NUMCON)
C
C**********************************************************************
C*     SUBROUTINE TO PERFORM CONTOUR PLOTS INPUT                      *
C**********************************************************************
C
      COMMON /PCONT/ ITITC(5,72),ITITXC(5,36),ITITYC(5,36),NC(5),
     *               NODESC(5,20)
C
      CALL HEADER1
      WRITE (6,4000)
C
      DO 100 I=1,NUMCON
      WRITE (6,2000) I
C
C     READ TITLES
C
      READ (5,1002) (ITITC(I,J),J=1,72)
      READ (5,1002) (ITITXC(I,J),J=1,36)
      READ (5,1002) (ITITYC(I,J),J=1,36)
      WRITE (6,2001) (ITITC(I,J),J=1,72)
      WRITE (6,2002) (ITITXC(I,J),J=1,36)
      WRITE (6,2003) (ITITYC(I,J),J=1,36)
C
C     READ NUMBER OF NODE POINTS
C
      READ (5,1000) NC(I)
      NCI=NC(I)
      IF (NCI.GT.0.AND.NCI.LE.20) GO TO 10
      WRITE (6,3000)
      WRITE (6,3001) I,NCI
      CALL ADIOS (2)
C
   10 READ (5,1000) (NODESC(I,J),J=1,NCI)
      WRITE (6,2004) I,NCI,(NODESC(I,J),J=1,NCI)
C
  100 CONTINUE
C
      RETURN
C
 1000 FORMAT (16I5)
 1002 FORMAT (72A1)
 2000 FORMAT (/////25X," CONTOUR GRAPH NO.  ",I5/
     * / 15X,"T I T L E S"/)
 2001 FORMAT (/ 15X,"GRAPH TITLE"
     * / 5X,72A1)
 2002 FORMAT (/20X,"X-AXIS TITLE"/10X,72A1)
 2003 FORMAT (/20X,"Y-AXIS TITLE"/10X,72A1)
 2004 FORMAT (/
     * 15X,"GRAPH NO. ",I5,5X,"NO. OF POINTS ",I5
     * / 15X,"POINTS =",20I5)
 3000 FORMAT (15X,"--- ERROR TERMINATION FROM PLTCIN---")
 3001 FORMAT (/15X,"NO. OF POINTS IS WRONG, GRAPH NO. =",I5/
     * / 30X,"NO OF POINTS = ",I5,5X"SHOULD BE > 0 AND < OR = 20")
 4000 FORMAT(/20X,"C O N T O U R   P L O T   I N P U T   E C H O"//)
C
      RETURN
C
      END
```

EXAMPLE PROBLEM 1: FLANKING LOSS - STATIC CASE

```
FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE
    2   -1    0    1    1    1    1    1    0    0
    0    0    0    0    0    0    0    0    0   -1
   -1   -1   -1   -1   -1
    4    3    4    5    4    3    4
     .254      .273      .400      .502      .578      0.610      .864
    5    3    5
     .456     .457      .957
      2.0        0.0       7.45      7.45       0.0       2.33       2.33       0.0
     25.0     100.       65.0      65.0      100.       38.        38.       100.
     0.022     0.430     0.180     0.180     .036
     32.00      1920.     1100.     1100.      15.10
     921.1     963.0     2000.     2000.      1214.
    1      38.0    2       38.2
      0.0      3600.      900.        1.0
     25.0
    1    1    0    6    1
      20.       80.
TEST PLOT FOR CONTOURS - RJO 09/01/82
X - AXIS   NODES
Y - AXIS   TEMPERATURE (C)
   20
    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
   17   18   19   20
 TEST PLOT - TIME HISTORIES 08/27/82 BY PJB
 X - AXIS   TIME (SEC)
 Y - AXIS   TEMPERATURE (C)
    5
    1    2    3    4    5
```

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

C O N T R O L   D A T A

NUMTBC	=	2
NUMIT	=	-1
IDATA	=	0
IPR	=	1
IRFINE	=	1
IDGT	=	1
IEB	=	1
IQPR	=	1
ITRAN	=	0
IPLT	=	0

DEBUG PARAMETERS

1 (WRITE A,C MATRICES. B VECTOR)	=	0
2 (WRITE DECOMPOSED C MATRIX)	=	0
3 (WRITE NODAL ASSOC.)	=	0
4 (WRITE ELEMENT MATRICES)	=	0
5 (WRITE INTERIOR DATA)	=	0
6 (WRITE C BEFORE DECOMP.)	=	0
7 (WRITE B VECTOR)	=	0
8 ( )	=	0
9 ( )	=	0
10( )	=	-1

E N E R G Y   B A L A N C E S

ICHECK(1)	=	-1
ICHECK(2)	=	-1
ICHECK(3)	=	-1
ICHECK(4)	=	-1
ICHECK(5)	=	-1

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

G R I D   S P E C I F I C A T I O N

Y NUMBERS FOLLOW

NY5	=	4
NY4	=	3
NY1L	=	4
NYWALL	=	5
NY1R	=	4
NY7	=	3
NY8	=	4

Y'S FOLLOW

Y5	=	0.2540E+00
Y4	=	0.2730E+00
Y1L	=	0.4000E+00
YWALL	=	0.5020E+00
Y1R	=	0.5780E+00
Y7	=	0.6100E+00
Y8	=	0.8640E+00

Z NUMBERS FOLLOW

NZPOLY	=	5
NZFRP	=	3
NZWALL	=	5

Z'S FOLLOW

ZPOLY	=	0.4560000E+00
ZFRP	=	0.4570000E+00
ZWALL	=	0.9570000E+00

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

C U M U L A T I V E   N O D E   N U M B E R S

Y - DIRECTION NODE NUMBERS		
NTY4	=	7
NTY1L	=	11
NTYWAL	=	16
NTY1R	=	20
NTY7	=	23
NCOL	=	27
Z - DIRECTION NODE NUMBERS		
NTZFRP	=	8
NTZWAL	=	13

TOTAL NUMBER OF NODES (EQUATIONS) = 246

LOWER NUMBER OF NODE POINTS = 216

'

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

C O N V E C T I V E   C O E F F I C I E N T S

HEAT TRANSFER COEFFICIENTS FOLLOW

0.20000E+01 0.00000E+00 0.74500E+01 0.74500E+01 0.00000E+00 0.23300E+01 0.23300E+01 0.00000E+00

AMBIENT TEMPERATURES FOLLOW

0.25000E+02 0.10000E+03 0.65000E+02 0.65000E+02 0.10000E+03 0.38000E+02 0.38000E+02 0.10000E+03

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

M A T E R I A L   P R O P E R T I E S

I	RKAY	RHO	CP
1	0.2200E-01	0.3200E+02	0.9211E+03
2	0.4300E+00	0.1920E+04	0.9630E+03
3	0.1800E+00	0.1100E+04	0.2000E+04
4	0.1800E+00	0.1100E+04	0.2000E+04
5	0.3600E-01	0.1510E+02	0.1214E+04
6	0.0000E+00	0.0000E+00	0.0000E+00

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

1

T E M P E R A T U R E   B O U N D A R Y   C O N D

NUMBER OF TEMPERATURE BCS = 2

1 0.380E+02 2 0.382E+02

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

I N I T I A L   T E M P E R A T U R E S

INITIAL TEMPERATURE = 0.00000E+00

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

--- INPUT PHASE COMPLETE ---

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

E Q U A T I O N   D A T A

DETERMINANT	=	0.1374913E+76
ACCURACY TEST PARAMETER	=	0.1924661E+06
NUMBER OF EQUATIONS	=	246
ROW SIZE IN CALLING ROUTINE	=	246
ERROR PARAMETER (129-SIN.,34-ACC.)	=	0

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

T E M P E R A T U R E   S T A T E

A T   T I M E   =   0.00000E+00

1	0.0000000E+00	51	0.0000000E+00	101	0.0000000E+00	151	0.0000000E+00	201	0.0000000E+00
2	0.0000000E+00	52	0.0000000E+00	102	0.0000000E+00	152	0.0000000E+00	202	0.0000000E+00
3	0.0000000E+00	53	0.0000000E+00	103	0.0000000E+00	153	0.0000000E+00	203	0.0000000E+00
4	0.0000000E+00	54	0.0000000E+00	104	0.0000000E+00	154	0.0000000E+00	204	0.0000000E+00
5	0.0000000E+00	55	0.0000000E+00	105	0.0000000E+00	155	0.0000000E+00	205	0.0000000E+00
6	0.0000000E+00	56	0.0000000E+00	106	0.0000000E+00	156	0.0000000E+00	206	0.0000000E+00
7	0.0000000E+00	57	0.0000000E+00	107	0.0000000E+00	157	0.0000000E+00	207	0.0000000E+00
8	0.0000000E+00	58	0.0000000E+00	108	0.0000000E+00	158	0.0000000E+00	208	0.0000000E+00
9	0.0000000E+00	59	0.0000000E+00	109	0.0000000E+00	159	0.0000000E+00	209	0.0000000E+00
10	0.0000000E+00	60	0.0000000E+00	110	0.0000000E+00	160	0.0000000E+00	210	0.0000000E+00
11	0.0000000E+00	61	0.0000000E+00	111	0.0000000E+00	161	0.0000000E+00	211	0.0000000E+00
12	0.0000000E+00	62	0.0000000E+00	112	0.0000000E+00	162	0.0000000E+00	212	0.0000000E+00
13	0.0000000E+00	63	0.0000000E+00	113	0.0000000E+00	163	0.0000000E+00	213	0.0000000E+00
14	0.0000000E+00	64	0.0000000E+00	114	0.0000000E+00	164	0.0000000E+00	214	0.0000000E+00
15	0.0000000E+00	65	0.0000000E+00	115	0.0000000E+00	165	0.0000000E+00	215	0.0000000E+00
16	0.0000000E+00	66	0.0000000E+00	116	0.0000000E+00	166	0.0000000E+00	216	0.0000000E+00
17	0.0000000E+00	67	0.0000000E+00	117	0.0000000E+00	167	0.0000000E+00	217	0.0000000E+00
18	0.0000000E+00	68	0.0000000E+00	118	0.0000000E+00	168	0.0000000E+00	218	0.0000000E+00
19	0.0000000E+00	69	0.0000000E+00	119	0.0000000E+00	169	0.0000000E+00	219	0.0000000E+00
20	0.0000000E+00	70	0.0000000E+00	120	0.0000000E+00	170	0.0000000E+00	220	0.0000000E+00
21	0.0000000E+00	71	0.0000000E+00	121	0.0000000E+00	171	0.0000000E+00	221	0.0000000E+00
22	0.0000000E+00	72	0.0000000E+00	122	0.0000000E+00	172	0.0000000E+00	222	0.0000000E+00
23	0.0000000E+00	73	0.0000000E+00	123	0.0000000E+00	173	0.0000000E+00	223	0.0000000E+00
24	0.0000000E+00	74	0.0000000E+00	124	0.0000000E+00	174	0.0000000E+00	224	0.0000000E+00
25	0.0000000E+00	75	0.0000000E+00	125	0.0000000E+00	175	0.0000000E+00	225	0.0000000E+00
26	0.0000000E+00	76	0.0000000E+00	126	0.0000000E+00	176	0.0000000E+00	226	0.0000000E+00
27	0.0000000E+00	77	0.0000000E+00	127	0.0000000E+00	177	0.0000000E+00	227	0.0000000E+00
28	0.0000000E+00	78	0.0000000E+00	128	0.0000000E+00	178	0.0000000E+00	228	0.0000000E+00
29	0.0000000E+00	79	0.0000000E+00	129	0.0000000E+00	179	0.0000000E+00	229	0.0000000E+00
30	0.0000000E+00	80	0.0000000E+00	130	0.0000000E+00	180	0.0000000E+00	230	0.0000000E+00
31	0.0000000E+00	81	0.0000000E+00	131	0.0000000E+00	181	0.0000000E+00	231	0.0000000E+00
32	0.0000000E+00	82	0.0000000E+00	132	0.0000000E+00	182	0.0000000E+00	232	0.0000000E+00
33	0.0000000E+00	83	0.0000000E+00	133	0.0000000E+00	183	0.0000000E+00	233	0.0000000E+00
34	0.0000000E+00	84	0.0000000E+00	134	0.0000000E+00	184	0.0000000E+00	234	0.0000000E+00
35	0.0000000E+00	85	0.0000000E+00	135	0.0000000E+00	185	0.0000000E+00	235	0.0000000E+00
36	0.0000000E+00	86	0.0000000E+00	136	0.0000000E+00	186	0.0000000E+00	236	0.0000000E+00
37	0.0000000E+00	87	0.0000000E+00	137	0.0000000E+00	187	0.0000000E+00	237	0.0000000E+00
38	0.0000000E+00	88	0.0000000E+00	138	0.0000000E+00	188	0.0000000E+00	238	0.0000000E+00
39	0.0000000E+00	89	0.0000000E+00	139	0.0000000E+00	189	0.0000000E+00	239	0.0000000E+00
40	0.0000000E+00	90	0.0000000E+00	140	0.0000000E+00	190	0.0000000E+00	240	0.0000000E+00
41	0.0000000E+00	91	0.0000000E+00	141	0.0000000E+00	191	0.0000000E+00	241	0.0000000E+00
42	0.0000000E+00	92	0.0000000E+00	142	0.0000000E+00	192	0.0000000E+00	242	0.0000000E+00
43	0.0000000E+00	93	0.0000000E+00	143	0.0000000E+00	193	0.0000000E+00	243	0.0000000E+00
44	0.0000000E+00	94	0.0000000E+00	144	0.0000000E+00	194	0.0000000E+00	244	0.0000000E+00
45	0.0000000E+00	95	0.0000000E+00	145	0.0000000E+00	195	0.0000000E+00	245	0.0000000E+00
46	0.0000000E+00	96	0.0000000E+00	146	0.0000000E+00	196	0.0000000E+00	246	0.0000000E+00
47	0.0000000E+00	97	0.0000000E+00	147	0.0000000E+00	197	0.0000000E+00		
48	0.0000000E+00	98	0.0000000E+00	148	0.0000000E+00	198	0.0000000E+00		
49	0.0000000E+00	99	0.0000000E+00	149	0.0000000E+00	199	0.0000000E+00		
50	0.0000000E+00	100	0.0000000E+00	150	0.0000000E+00	200	0.0000000E+00		

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

T E M P E R A T U R E   S T A T E

A T   T I M E = 0.10000E+01

1	0.3800000E+02	51	0.3522160E+02	101	0.5312222E+02	151	0.6328056E+02	201	0.4586673E+02
2	0.3820001E+02	52	0.3524992E+02	102	0.5316809E+02	152	0.6422050E+02	202	0.5002994E+02
3	0.2653259E+02	53	0.3528310E+02	103	0.5320329E+02	153	0.6447458E+02	203	0.5428506E+02
4	0.2618610E+02	54	0.3529651E+02	104	0.5322792E+02	154	0.6441223E+02	204	0.5868507E+02
5	0.2621045E+02	55	0.3572284E+02	105	0.5394386E+02	155	0.6389954E+02	205	0.6330580E+02
6	0.2621477E+02	56	0.3547023E+02	106	0.5432954E+02	156	0.6385086E+02	206	0.6422902E+02
7	0.2619913E+02	57	0.3483316E+02	107	0.5451755E+02	157	0.6388768E+02	207	0.6448042E+02
8	0.2573653E+02	58	0.3465353E+02	108	0.5457301E+02	158	0.6401218E+02	208	0.6441872E+02
9	0.2562080E+02	59	0.3465717E+02	109	0.3790329E+02	159	0.6469182E+02	209	0.6391583E+02
10	0.2561020E+02	60	0.3466478E+02	110	0.3789857E+02	160	0.6473489E+02	210	0.6388184E+02
11	0.2563477E+02	61	0.3467639E+02	111	0.3789522E+02	161	0.6474092E+02	211	0.6391768E+02
12	0.2565886E+02	62	0.3528627E+02	112	0.3787305E+02	162	0.6473187E+02	212	0.6402451E+02
13	0.2568828E+02	63	0.3600941E+02	113	0.3787134E+02	163	0.3790366E+02	213	0.6469533E+02
14	0.2572438E+02	64	0.3685211E+02	114	0.3787346E+02	164	0.3789894E+02	214	0.6473795E+02
15	0.2577219E+02	65	0.3780980E+02	115	0.3787935E+02	165	0.3789558E+02	215	0.6474391E+02
16	0.2584421E+02	66	0.3847417E+02	116	0.3808525E+02	166	0.3787350E+02	216	0.6473492E+02
17	0.2595937E+02	67	0.3916658E+02	117	0.3842395E+02	167	0.3787259E+02	217	0.4140094E+02
18	0.2617430E+02	68	0.3987474E+02	118	0.3927539E+02	168	0.3787469E+02	218	0.4588406E+02
19	0.2660481E+02	69	0.4058566E+02	119	0.4175702E+02	169	0.3787997E+02	219	0.5036676E+02
20	0.2750252E+02	70	0.4128700E+02	120	0.4586623E+02	170	0.3808499E+02	220	0.5486147E+02
21	0.2755939E+02	71	0.4192239E+02	121	0.5002855E+02	171	0.3842264E+02	221	0.5937775E+02
22	0.2753908E+02	72	0.4253560E+02	122	0.5428287E+02	172	0.3927151E+02	222	0.6392088E+02
23	0.2744057E+02	73	0.4312404E+02	123	0.5868193E+02	173	0.4174657E+02	223	0.4140939E+02
24	0.2607216E+02	74	0.4368668E+02	124	0.6326938E+02	174	0.4586659E+02	224	0.4591107E+02
25	0.2592133E+02	75	0.4372028E+02	125	0.6421651E+02	175	0.5002953E+02	225	0.5041364E+02
26	0.2590688E+02	76	0.4374571E+02	126	0.6447176E+02	176	0.5428441E+02	226	0.5491788E+02
27	0.2590627E+02	77	0.4376297E+02	127	0.6440915E+02	177	0.5868411E+02	227	0.5942415E+02
28	0.3561974E+02	78	0.4426237E+02	128	0.6389107E+02	178	0.6329269E+02	228	0.6393213E+02
29	0.3493483E+02	79	0.4456305E+02	129	0.6383527E+02	179	0.6422467E+02	229	0.4141129E+02
30	0.3215359E+02	80	0.4472310E+02	130	0.6387257E+02	180	0.6447746E+02	230	0.4591534E+02
31	0.3099481E+02	81	0.4477321E+02	131	0.6400600E+02	181	0.6441542E+02	231	0.5041951E+02
32	0.3098889E+02	82	0.3674188E+02	132	0.6469005E+02	182	0.6390779E+02	232	0.5492386E+02
33	0.3098637E+02	83	0.3670261E+02	133	0.6473334E+02	183	0.6386638E+02	233	0.5942836E+02
34	0.3098721E+02	84	0.3668343E+02	134	0.6473940E+02	184	0.6390271E+02	234	0.6393298E+02
35	0.3113385E+02	85	0.3701643E+02	135	0.6473035E+02	185	0.6401834E+02	235	0.4141151E+02
36	0.3137706E+02	86	0.3702560E+02	136	0.3790347E+02	186	0.6469357E+02	236	0.4591586E+02
37	0.3170750E+02	87	0.3703944E+02	137	0.3789876E+02	187	0.6473643E+02	237	0.5042020E+02
38	0.3211189E+02	88	0.3705801E+02	138	0.3789540E+02	188	0.6474242E+02	238	0.5492451E+02
39	0.3240359E+02	89	0.3797911E+02	139	0.3787328E+02	189	0.6473340E+02	239	0.5942880E+02
40	0.3271690E+02	90	0.3910089E+02	140	0.3787199E+02	190	0.3790384E+02	240	0.6393307E+02
41	0.3304875E+02	91	0.4051501E+02	141	0.3787410E+02	191	0.3789912E+02	241	0.4141150E+02
42	0.3339658E+02	92	0.4230946E+02	142	0.3787968E+02	192	0.3789578E+02	242	0.4591582E+02
43	0.3375835E+02	93	0.4366922E+02	143	0.3808513E+02	193	0.3787375E+02	243	0.5042014E+02
44	0.3410638E+02	94	0.4512498E+02	144	0.3842332E+02	194	0.3787315E+02	244	0.5492444E+02
45	0.3446364E+02	95	0.4661453E+02	145	0.3927354E+02	195	0.3787524E+02	245	0.5942876E+02
46	0.3482692E+02	96	0.4807436E+02	146	0.4175206E+02	196	0.3788023E+02	246	0.6393306E+02
47	0.3518811E+02	97	0.4943433E+02	147	0.4586642E+02	197	0.3808484E+02		
48	0.3520857E+02	98	0.5053716E+02	148	0.5002907E+02	198	0.3842191E+02		
49	0.3522148E+02	99	0.5149931E+02	149	0.5428368E+02	199	0.3926932E+02		
50	0.3522702E+02	100	0.5235011E+02	150	0.5868307E+02	200	0.4174054E+02		

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

E N E R G Y   B A L A N C E S   A T

T I M E   =   0.1000000E+01

T O T A L   L I N E   Q ' S

HT (1)	=-0.3924801E+01
HT (2)	=-0.1461942E+00
HT (3)	=-0.9480131E+00
HT (4)	= 0.1400995E+01
HT (5)	= 0.4217818E+01
HT (6)	=-0.4009733E+01
HT (7)	=-0.1973910E+00
HT (8)	=-0.5387869E+00
HT (9)	=-0.5843179E+00
HT (10)	=-0.9703678E-01
HT (11)	=-0.1521804E+00
HT (12)	=-0.8679169E-01
HT (13)	=-0.1501728E+00

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

E N E R G Y   B A L A N C E S

H1 (REGION 5-6-4U-4L-1L-2L)	=-0.3496613E+01
H2 (REGION 1C)	= 0.5151719E-01
H3 (REGION 1R-2R-7L-7U-8-9)	=-0.2283725E+00
H4 (REGION 2C)	= 0.8237422E-02
H5 (REGION 3)	= 0.5791229E-01

NET MC ENERGY =-0.1147186E+03

FRAME COEFFICIENTS-REL

HTW	= 0.1107266E+03
FCMCA	= 0.2621585E-01
FCHXS	= 0.3479850E-01

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

1

****

FLANK LOSS POLYSTYRENE B-B 9-01-82 TEMPB-STATIC CASE

****

--- N O R M A L   T E R M I N A T I O N ---

```
01/16/85  13:53:17
USER  CPU TIME        7:01.843    99.5 %
SVC   CPU TIME           2.082      .4 %
ROLL  CPU TIME           0.000     0.0 %
PROCESSOR TIME        7:03.925   100.0 %     53.7 %
WAIT      TIME        6:04.075               46.2 %
ROLL      TIME           0.000                0.0 %
ELAPSED   TIME       13:08                  100.0 %
ROLLS           0
I/O             677
```

EXAMPLE PROBLEM 2: FLANKING LOSS - DYNAMIC CASE

```
 FLANK LOSS POLYSTYRENE  09-01-82   TEMPB DYNAMIC CASE
    2   -1     0    4     1     1     1     8     1     0
    0    0     0    0     0     0     0     0     0    -1
   -1   -1    -1   -1    -1
    4    3     4    5     4     3     4
     0.254     0.273     0.400     0.502     0.578     0.610     0.864
    5    3     5
     0.456    0.457      .957
       2.0       0.0      7.45      7.45       0.0      2.33      2.33       0.0
      25.0      100.      65.0      65.0      100.      38.0      38.0      100.
     0.022     0.430     0.180     0.180    0.0360
     32.00     1920.     1100.     1100.     15.10
     921.1     963.0     2000.     2000.     1214.
    1      38.0    2      38.2
       0.0    86400.      900.       1.0
      25.0
    1    1     0    6     1
       20.       80.
TEST PLOT FOR CONTOURS - RJO 09/01/82
X - AXIS   NODES
Y - AXIS   TEMPERATURE (C)
   20
    1    2     3    4     5     6     7     8     9    10    11    12    13    14    15    16
   17   18    19   20
 TEST PLOT - TIME HISTORIES 08/27/82 BY RJO
 X - AXIS   TIME (SEC)
 Y - AXIS   TEMPERATURE (C)
    5
    1    2     3    4     5
```

NBS-114A (REV. 9-78)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO.	2. Gov't. Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE	5. Publication Date
Thermal Flanking Loss Calculations for the National Bureau of Standards Calibrated Hot Box	6. Performing Organization Code
7. AUTHOR(S) Ronald J. Onega and Patrick J. Burns	8. Performing Organ. Report No.
9. PERFORMING ORGANIZATION NAME AND ADDRESS NATIONAL BUREAU OF STANDARDS DEPARTMENT OF COMMERCE WASHINGTON, DC 20234	10. Project/Task/Work Unit No. 11. Contract/Grant No.
12. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS *(Street, City, State, ZIP)* Department of Energy National Bureau of Standards	13. Type of Report & Period Covered 14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

16. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)*

A two-dimensional, finite-difference model was developed to calculate the flanking loss for the NBS Calibrated Hot Box. A new definition of flanking loss is presented, along with the thermal theory, a description of the computer code and some results. This model applies to both steady-state and dynamic boundary conditions.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*

calibrated hot box; dynamic; finite difference; flanking loss

18. AVAILABILITY ☒ Unlimited	19. SECURITY CLASS (THIS REPORT)	21. NO. OF PRINTED PAGES
☐ For Official Distribution. Do Not Release to NTIS	UNCLASSIFIED	
☐ Order From Sup. of Doc., U.S. Government Printing Office, Washington, DC 20402, SD Stock No. SN003-003-	20. SECURITY CLASS (THIS PAGE)	22. Price
☐ Order From National Technical Information Service (NTIS), Springfield, VA. 22161	UNCLASSIFIED	

USCOMM-DC