2 mit

CR-134121

THERMAL AND FLOW ANALYSIS SUBROUTINES FOR THE SINDA-VERSION 9 COMPUTER ROUTINE

REPORT NO. 00.1582

24 September 1973

(NASA-CR-134121) THERMAL AND FLOW ANALYSIS SUBROUTINES FOR THE SINDA-VERSION 9 COMPUTER ROUTINE (LTV Aerospace Corp.) 274 p HC $15.75 CSCL 20D

N74-12045

Unclas

G3/12 22832

VOUGHT SYSTEMS DIVISION

LTV AEROSPACE CORPORATION

Performed Under
NASA-JSC Contract
NAS9-6807

THERMAL AND FLOW ANALYSIS SUBROUTINES FOR THE SINDA-VERSION 9 COMPUTER ROUTINE

Report No. 00.1582

24 September 1973

Submitted by

Vought Systems Division
LTV Aerospace Corporation
P.O. Box 5907
Dallas, Texas

To

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas

Prepared by:

J. A. Oren

D. R. Williams

Approved by:

R. J. French, Supervisor
Environmental Control/
Life Support Systems

TABLE OF CONTENTS

APPENDICES

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY AND INTRODUCTION

During the past decade extensive capabilities for combined thermal and fluid flow transient analysis was developed at the Vought Systems Division (VSD) of LTV Aerospace Corporation. The capabilities included (1) a pressure/flow solution for a general flow network (integrated with the finite difference temperature solution) including general valve analyses, orifice and pump analysis packages, (2) a number of special thermal analysis options including heat exchanger analysis, cavity radiant interchange analysis, cabin analysis, etc. and (3) a number of input/output capabilities such as automatic plotting, interrupt and restart, etc. These capabilities were included in a general purpose thermal analysis routine, MOTAR[1*], developed by VSD for NASA-JSC.

The objective of the effort described by this report was to incorporate these fluid flow analysis, special thermal analysis and input/output capabilities of the MOTAR routine into the SINDA[7] routine which was developed by the TRW Corporation. This effort was performed under contract NAS9-6807 for NASA-JSC. All the capabilities were added in the form of user subroutines so that they may be added to different versions of SINDA with a minimum of programmer effort.

Two modifications were made to the existing subroutines of SINDA/VERSION 8 to incorporate the above subroutines. These were:

(1) A modification to the preprocessor to permit actual values of array numbers, conductor numbers, node numbers or constant numbers supplied as array data to be converted to relative numbers.

(2) Modifications to execution subroutine CNFAST to make it compatible with the radiant interchange user subroutine, RADIR.

This modified version of SINDA has been designated SINDA/VERSION 9.

A detailed discussion of the methods used for the capabilities added is presented in Section 2.0. The modifications for the SINDA subroutines are described in Section 3.0. User subroutines are described in Section 4.0, and a sample problem is given in Section 5.0. All subroutines added or modified are listed in Appendix A.

---

* Superscripts refer to references in Section 6.0

2.0 DISCUSSION OF METHODS

SINDA user subroutines were developed to incorporate the MOTAR routine[1] capabilities for fluid/pressure analysis, thermal analysis of a flowing fluid and enclosure analysis into the SINDA routine. The analytical methods for these capabilities are described in this section. Thermal analysis features such as those methods required for analysis of a flowing fluid and those required for enclosure radiation analysis are described in Section 2.1. Pressure-flow analysis methods are described in Section 2.2.

2.1 Thermal Analysis Features

The calculation methods for (1) convection and flow thermal conductors for flow in a tube, (2) heat exchanger thermal performance, (3) inline heater thermal performance, (4) cabin thermal and mass balance, and (5) enclosure radiation thermal performance have been added to the SINDA library of user subroutines. The methods used are based on those from the MOTAR computer routine and are described in detail in the following sections.

2.1.1 Convection Conductors

Three user subroutines were prepared for the SINDA library to give to the user the capability of analyzing convection heat transfer for flow in a tube. These subroutines and their functions are:

- CONV1 - Calculates heat transfer coefficient using relationships for convection in a flowing tube
- CONV2 - Calculates the heat transfer coefficient using the Stanton number obtained from interpolating a curve
- CONV3 - Interpolates a curve of heat transfer coefficients vs flowrates for the coefficient

The value of the convection conductor, G_{ij}, between a fluid lump and tube lump is given by the following relation for all three of the above routines:

$$G_{ij} = hA \tag{1}$$

where

- h - the convection heat transfer coefficient
- A - the convection area

CONV1 uses one of several methods for determining the heat transfer coefficient, h, for flowing fluid in a tube depending on the flow regime. The flow regime is assumed to be laminar when the Reynolds number is 2000 or less. For this regime the convection heat transfer coefficient is calculated by:

$$h = \frac{k}{D}\left[3.66 \cdot F1 + \frac{.0155 \cdot F2}{\frac{1}{R_e P_r}\ \frac{X}{D} + .015\left[\frac{1}{R_e P_r}\ \frac{X}{D}\right]^{1/3}}\right] \tag{2}$$

where:

- k = thermal conductivity
- D = hydraulic diameter to flow
- X = distance from tube entrance
- Re = Reynolds number $= \frac{4\,\dot{m}}{\mu P}$
- $\dot{m}$ = flow rate of fluid
- μ = viscosity of fluid
- P = wetted perimeter of fluid flow passage
- $F1$ = An input factor for modifying fully developed flow
- $F2$ = An input factor for modifying developing flow

Equation (2) is a curve fit obtained by VMSC to approximate the Gratz solution[2] to flow in a tube for values of $\frac{X}{D}\ \frac{1}{RePr}$ greater than 0.001.

The convection heat transfer coefficient for flow in a tube in the transition flow regime (2000 < Re < 6400) is approximated in CONV1 by the following relation:

$$h = \frac{K}{D}\left[0.116\,(Re^{2/3} - 125)(Pr)^{1/3}\right] \tag{3}$$

This relation was derived by Hausen[3] and holds only for fully developed flow.

The relation used in CONV1 to determine h for turbulent flow (Re $\geq$ 6400) is the following:

$$h = .023 \frac{K}{D} (Re)^{.8} (Pr)^{1/3} \quad (4)$$

CONV2 supplies a more general option for determining the convection heat transfer coefficient. A curve of $St(Pr)^{2/3}$ vs Reynolds No. is interpolated to obtain the value of $St(Pr)^{2/3}$. That is,

$$St(PR)^{2/3} = F(Re) \quad (5)$$

Where St = Stantion number

$$= \frac{Nu}{Re\ Pr}$$

$$= \frac{h}{CpV}$$

V = Average fluid velocity

$F(Re)$ = An arbitrary function of Reynonds number which the user can input as a table

The heat transfer coefficient is calculated by

$$h = \frac{K}{D} F(Re)\ Re(Pr)^{1/3} \quad (6)$$

In CONV3, the convection heat transfer coefficient is obtained by direct interpolation of a curve of heat transfer coefficient vs flowrate.

2.1.2 Flow Conductors

A method for calculating the value of flow conductors is required when analyzing a problem with fluid flowing in a tube. The flow conductor is a one way conductor from node i to node j and is calculated by

$$G_{ij} = \dot{W}\ Cp_i \quad (7)$$

where:
G_{ij} = the conductance from the upstream lump
$\dot{W}$ = the mass flow rate in the tube
Cp_i = the fluid specific heat for lump i

Two user subroutines FLOCN1 and FLOCN2, were prepared to calculate the values for the flow conductors. Both subroutines reference the flowrate array and an array containing conductor identification information. FLOCN1 assumes the specific heat is a function of temperature whereas; FLOCN2 assumes a constant value for specific heat.

2.1.3 Heat Exchanger Analysis

Four subroutines have been written to facilitate the thermal analysis of systems containing heat exchangers. These are HXCNT for analysis of counter flow heat exchangers, HXPAR for parallel flow heat exchangers, HXCROS for cross flow heat exchangers and HXEFF for any heat exchanger with an input effectiveness. These subroutines calculate the outlet temperatures of two sides based upon the inlet temperatures and heat exchanger effectiveness. The relations used for calculating effectiveness are described below.

2.1.3.1 Counterflow Heat Exchanger

Subroutine HXCNT calculates the heat exchanger effectiveness using the relation from Reference 3 for counterflow heat exchangers. That is,

$$\epsilon = \frac{1-e^{-\left[\frac{UA}{(MC)_s}\left\{1-\frac{(MC)_s}{(MC)_l}\right\}\right]}}{1-\frac{(MC)_s}{(MC)_l}\;e^{-\left[\frac{UA}{(MC)_s}\left\{1-\frac{(MC)_s}{(MC)_l}\right\}\right]}} \qquad (8)$$

Where ϵ = effectiveness

UA = overall effectiveness

$(MC)_s$ = mass, specific heat product for the side with the smallest MC

$(MC)_l$ = mass, specific heat product for the side with the largest MC

The limiting cases for this relation are:

(1) When $(MC)_s/(MC)_l = 0$,

$$\epsilon = 1 - e^{-UA/(MC)_s}$$

(2) When $(MC)_s/(MC)_l = 1$

$$\epsilon = \frac{\frac{UA}{(MC)_s}}{1+\frac{UA}{(MC)_s}} = \frac{UA}{(MC)_s + UA}$$

Using the effectiveness as calculated by the above method, the outlet temperatures are calculated as follows:

1. For the side with the smallest MC, $(MC)_s$:

$$Tout_s = Tin_s - \epsilon\,(Tin_s - Tin_l) \qquad (9)$$

2. The outlet temperature for the side with the large MC is then calculated by

$$Tout_l = \frac{(MC)_s}{(MC)_l}\,(Tin_s - Tout_s) + Tin_l \qquad (10)$$

2.1.3.2 Parallel Flow Heat Exchanger

Subroutine HXPAR calculates the heat exchanger effectiveness using the relation for parallel flow heat exchangers[3] which is:

$$\epsilon = \frac{1 - e^{-\frac{UA}{(MC)_s}\left[1+\frac{(MC)_s}{(MC)_l}\right]}}{1 + \frac{(MC)_s}{(MC)_l}} \qquad (11)$$

The limiting cases are

(1) When $(MC)_s/(MC)_l = 0$,

$$\epsilon = 1 - e^{-UA/(MC)_s}$$

(2) When $(MC)_s/(MC)_l = 1.$,

$$\epsilon = \frac{1 - e^{-2\frac{UA}{(MC)_s}}}{2.0}$$

The heat exchanger outlet temperatures are then calculated using equations 9 and 10.

2.1.3.3 Cross Flow Heat Exchanger

Subroutine HXCROS calculates the effectiveness for cross flow heat exchangers using one of the four relations below depending upon mixing of the streams.

Both Streams Unmixed

$$\epsilon = 1 - e^{\left[\left(e^{\left[-\frac{UA}{(MC)_s}\frac{(MC)_s}{(MC)_l}\eta\right]}-1\right)\frac{(MC)_l}{(MC)_s}\frac{1}{\eta}\right]} \tag{12}$$

Where $\eta = \left[\frac{(MC)_s}{UA}\right]^{0.22}$

Both Streams Mixed

$$\epsilon = \frac{\frac{UA}{(MC)_s}}{\dfrac{\frac{UA}{MC_s}}{1-e^{-\frac{UA}{(MC)_s}}} + \dfrac{\frac{UA}{(MC)_l}}{1-e^{-\frac{UA}{(MC)_l}}}} \tag{13}$$

Stream $(MC)_s$ Unmixed

$$\epsilon = \frac{1 - e^{-\frac{(MC)_s}{(MC)_l}\left[1 - e^{-\frac{UA}{(MC)_s}}\right]}}{\frac{(MC)_s}{(MC)_l}} \tag{14}$$

Stream $(MC)_l$ Unmixed

$$\epsilon = 1 - e^{-\frac{(MC)_l}{(MC)_s}\left[1 - e^{-\frac{UA}{(MC)_l}}\right]} \tag{15}$$

The heat exchanger outlet temperatures are calculated using equations (9) and (10).

2.1.3.4 User Supplied Effectiveness

Subroutine HXEFF was written to perform heat exchanger thermal analysis with a user supplied effectiveness. The effectiveness may either be supplied as a constant or as an array number which gives the effectiveness as a bivariant function of the flowrates on the two sides. The outlet temperatures are then calculated using equations (9) and (10).

2.1.4 Inline Heater Analysis

Provisions for the analysis of a fluid heater have been included in SINDA with subroutine HEATER. This subroutine simulates an electrical heater with a control system which turns the heater on when a specified sensor lump drops below a set value and turns the heater off when the specified sensor lump rises above another set value. When the heater is on an input quantity of heat is added to the heater node.

2.1.5 Cabin Analysis

A subroutine has been written for use with SINDA which will give the user the ability to perform thermal analyses on cabin air systems including condensation on the walls and a vapor mass balance. The cabin heat transfer and condensation analysis involves the two-component flow of a condensible vapor and a non-condensable gas, with condensation of the vapor occurring on surfaces in contact with the fluid. Two problems of this nature have been studied extensively.

1. Condensation on, or evaporation from, a surface over which a free stream of fluid is passing. In this case, for relatively low mass transfer rates, the fluid properties can be assumed to be constant.
2. Dehumidification of a confined fluid stream by a bank of tubes. In this case there is a marked change in the temperature and vapor content of the fluid, and the detailed deposition of the condensate is not of primary interest. This type of analysis is usually handled on an overall basis similar to heat exchanges effectiveness calculations.

The following additional assumptions have been made with respect to the cabin atmospheric conditions.

1. The heat of circulation in the cabin is sufficiently high that the temperature and humidity are effectively the same throughout the cabin.
2. The velocity at all points where heat transfer and/or condensation can occur is known, and is proportional to the total mass flow rate in the cabin.

These assumptions make it possible to calculate the heat and vapor balance in the cabin for the entire volume as a unit, and to solve the heat transfer and condensation equations at each node independently of the other nodes.

Cabin humidity can be determined from an overall vapor balance in the cabin. The total vapor in the cabin at the end of an iteration is:

$$W_V = W_V^{\,i-1} + W_{V\,in} - W_{V\,out} - \Sigma W_L$$

Where
W_V = mass of vapor in cabin at end of iteration i
$W_V^{\,i-1}$ = mass of vapor in cabin at start of iteration i-1
$W_{V\,in}$ = mass of vapor flowing into cabin during iteration i
$W_{V\,out}$ = mass of vapor flowing out of cabin during iteration i
ΣW_L = mass of vapor condensed during iteration i-1

$W_{V\,in}$ is determined from the known conditions of the gas flowing into the cabin.

$$W_{V\,in} = \dot{m}\,in \left[\frac{\psi in}{1. + \psi in} \right]$$

Where
$\dot{m}$ in = mass flow rate into cabin
ψ in = specific humidity of gas flowing into cabin
= time increment

It is assumed that an equal volume of gas is flowing out of the cabin. Then,

$$W_{V\,out} = \dot{m}\,out \left[\frac{\psi_c}{1 + \psi_c} \right]$$

Where ψ_c = specific humidity in the cabin (at the end of the previous iteration)

and $\dot{m}$ out = $\dot{m}$ in $[\rho_c / \rho in]$

Where
ρ_c = cabin density
ρ in = density of gas flowing into cabin

The condensation term ΣW_L is determined from the calculations for the individual nodes as described below. The properties of the cabin atmosphere are determined from the calculated value of W_V. The vapor pressure in the cabin is

$$P_v = \frac{W_v}{V_c} R_v T_c$$

Where V_c = cabin volume
R_v = gas constant
T_c = temperature of cabin gas
P_v = vapor pressure

Assuming that the cabin pressure P_c is a constant, the gas partial pressure P_a is:

$$P_a = P_c - P_v$$

and $$W_a = \frac{P_a}{R_a T_c}$$

Where W_a = mass of non-condensible gas in the cabin.

Now the new value of specific humidity in the cabin can be determined by

$$\psi_c = \frac{W_v}{W_a}$$

The properties of the atmosphere can now be determined by

$$\mu c = \frac{\chi \mu g + \psi c \mu v}{\chi + \psi c}$$

$$Cpc = \frac{Cpg + \psi cCpv}{1 + \psi c}$$

$$k_c = \frac{\chi kg + \psi ckv}{\chi + \psi c}$$

$$\rho c = \frac{W_v + W_s}{V_c}$$

Where μ = viscosity
C_p = specific heat
k = thermal conductivity
χ = molecular weight ratio, $\frac{M_v}{M_g}$

and all values are evaluated at $T_c^{\,i-1}$. Cabin temperature T_c can be determined by a heat balance on the cabin atmosphere.

$$T_c = T_c^{\,i-1} + \frac{\dot{m}_{in} C_{pc} \left(T_{in} - T_c^{\,i-1}\right) - \Sigma Q_L}{(W_V + W_A)\ C_{pc}}$$

Where $T_c^{\,i-1}$ = T_c after previous iteration

T_{in} = temperature of gas flowing into cabin

ΣQ_L = net heat loss to cabin lumps

The heat transfer between the cabin atmosphere and the tube and structure lumps in the cabin is defined by:

$$Q_{Li} = hA_{Li} \left[Tc - T_{Li} \right] \Delta\tau$$

Where h = heat transfer coefficient

A_{Li} = heat transfer area of lump

T_{Li} = temperature of tube lump

$\Delta\tau$ = time increment

Using the Colburn-Chilton heat transfer-mass transfer analogy, the condensation (or evaporation) at the tube lump is determined by:

$$\Delta W_{Li} = K_m A_{Li} \left[P_V - P_{Wi} \right] \Delta\tau$$

Where W_{Li} = condensation on wall, lb.

K_m = mass transfer coefficient

P_{Wi} = vapor pressure at T_{Li}

The latent heat addition to the lump due to this condensation is

$$\Delta Q_\lambda = \Delta W_{Li} \lambda$$

Where λ = latent heat of vaporization

The vapor pressure P_{Wi} can be determined by a relationship derived from the Clausius-Clapeyron equation and the perfect gas law (Appendix K of Reference 4).

$$P_{wi} = P_o \, e^{\frac{\lambda}{R_g T_o}\left[\frac{T_{Li} - T_o}{T_{Li}}\right]}$$

Where P_o is known vapor pressure at a reference temperature T_o.

Three methods are available for determining mass and heat transfer coefficient. For tube lumps the equations from Reference 3 for gas flowing normal to the tube axis was assumed. Three different equations are used depending on the value of the Reynold's number.

$$Nu = 0.43 + .533 \, (Re)^{.5} \, (Pr)^{.31} \quad Re < 4000$$

$$Nu = 0.43 + .193 \, (Re)^{.618} \, (Pr)^{.31} \quad 4000 < Re < 40000$$

$$Nu = 0.43 + .0265 \, (Re)^{.805} \, (Pr)^{.31} \quad 40000 < Re < 400000$$

These equations were derived for an air-vapor mixture, but should be relatively accurate for other similar gases. The Nusselt and Reynold's numbers in the equations are defined using the tube diameter for the characteristic dimension, and the velocity in the Reynold's number is input at each lump and ratioed to the total cabin atmosphere flow rate.

$$\nu i = \nu io \; \frac{\dot{W}c}{\dot{W}co}$$

Where

$\dot{W}co$ = nominal cabin atmosphere circulation rate
νio = velocity at lump at Wco
$\dot{W}_c$ = circulation rate at time of calculation

The second option assumes flat plate flow for cabin wall lumps. In this case the heat transfer coefficient, for laminar flow, varies along the plate. Hence, direction of gas flow and the location of an assumed leading edge must be assumed. The equation for flat plates from Reference 3 is:

$$N_u = 0.332 \, Re^{.5} \, Pr^{1/3}$$

where the Nusselt and Reynold's numbers are local values and are defined by the distance X from the assumed leading edge. For a wall lump of length L_i which is located a distance L_{io} from the assumed leading edge, the average Nusselt number can be defined as:

$$N_u = 0.664 \; Pr^{1/3} \left[(Re_1)^{.5} - (Re_o)^{.5} \right]$$

Where N_u is defined by L_i

Re_o is defined by L_{io}

Re_1 is defined by $L_{io} + L_i$

The third option is a direct user input for convective heat transfer coefficient.

For the determination of mass transfer coefficients, the same equations which were used for heat transfer coefficient can be used with the Sherwood number substituted for Nusselt number and Schmidt number for Prandtl number. However, if the diffusion coefficient for the cabin is approximately equal to thermal diffusivity, the Sherwood number is equal to the Nusselt number and the mass transfer coefficient can be determined directly from the heat transfer coefficient. That is:

$$Sh = Nu$$

$$\frac{K_m R T_g x}{D} = \frac{h x}{k}$$

If $D \cong \alpha$ then

$$K_m = \frac{hD}{\alpha \rho C_p RT_g} \tag{16}$$

$$K_m \cong \frac{h}{C_p P_c}$$

Equation (16) is the Lewis relationship (Reference 3). For a mixture of oxygen and water vapor characteristic values are .866 for the diffusion coefficient, D, and .879 for thermal diffusivity, α, so the relationship should be valid.

For cabin tube and wall lumps the values for ΔQ_{Li} and $\Delta Q_{\lambda i}$ are added to the basic heat balance equation for these lumps. Values for ΔQ_{Li}

are summed for all participating lumps for input to the cabin atmosphere heat balance. Values for ΔWL_i are also summed for all lumps for cabin humidity balance, and the value for total water condensed on each lump WL_i is maintained.

If the rate of evaporation or condensation is high it would be possible for the cabin humidity to change significantly during a single iteration. This could lead, for example, to overestimating condensation by assuming that the humidity is constant in the calculation. A test of the approximate vapor pressure in the cabin at the end of the iteration is made, and the condensation or evaporation at any lump is reduced, if the sign of the ΔWL_i term is changed. A value W_v' is calculated by:

$$W_v' = W_v^{i-1} - \Sigma W_L i$$

and

$$P_v' = \frac{W_v'}{144 V_c} R_v T_g$$

Then for each lump if

$$\frac{P_v' - P_{wi}}{P_v - P_{wi}} < 0$$

a new value of $\Delta W_L i$ is calculated by:

$$\Delta W_{Li} = \Delta W_{Li} \left[\frac{P_v - P_{wi}}{P_v - P_v'} \right]$$

The new values of ΔW_{Li} are now again summed for the new value of $\Sigma \Delta W_L$ for establishing cabin humidity for the next iteration. A test is also made to assure that W_v' is never less than zero.

2.1.6 Radiation Interchange Analysis

Capabilities have been incorporated into subroutines for use with SINDA to facilitate the analysis of radiation heat transfer in an enclosure. The capabilities include the ability to:

(1) Analyze diffuse and/or specular infrared radiation in an enclosure

(2) Analyze diffuse and/or specular radiation from an external source for as many wave bands as desired

(3) Consolidate several temperature nodes into a single surface to improve computational efficiency

A radiation surface is defined as a group of temperature nodes which may be assumed to have identical radiating properties, angle factors and interchange factors.

The subroutines account for the net radiation heat transfer between a number of surfaces due to emitted radiation from each surface, reflected radiation from each surface, and radiation from any number of incident sources. The reflection of the energy originally emitted by another surface or from an external source may be either diffuse, specular, or any combination of the two.

2.1.6.1 Emitted Radiation In A Cavity

The radiosity of a surface is defined as the flux of infrared radiation leaving that surface with a diffuse distribution (according to Lambert's Law). That energy leaving a surface which has been reflected in a specular manner does not contribute to the radiosity of that surface. The incident infrared radiosity is denoted by the symbol H. The reflectance $(1 - \varepsilon)$ of a surface is separated into two components, the diffuse reflectance (ρ), and the specular reflectance (ρ^s). Here ε is the emittance of the surface and is equivalent to the absorptance for long wavelength radiation. With the angle factors (F_{ij}) defined in the normal way, there exist similar angle factors which relate the geometrical ability of surface i to radiate to surface j by means of a mirror-like reflection from specular surface k. Reference to Figure 1 indicates the method of imagery which will enable the calculation of these reflected angle factors. Here the angle factor to surface j is identical with the angle factor to the image of surface j. Also the angle factor is limited by the ability of surface i to "see" through the "window" of surface k. With the specular surface angle factors so defined, an interchange factor E_{ij} is defined similarly to reference 5 as follows:

$$E_{ij} = \sum_k \rho_k^s \; F_{ij\,(k)} + \sum_k \sum_l (\rho_k^s)(\rho_l^s)\, F_{ij(k,l)} + \cdots \quad (17)$$

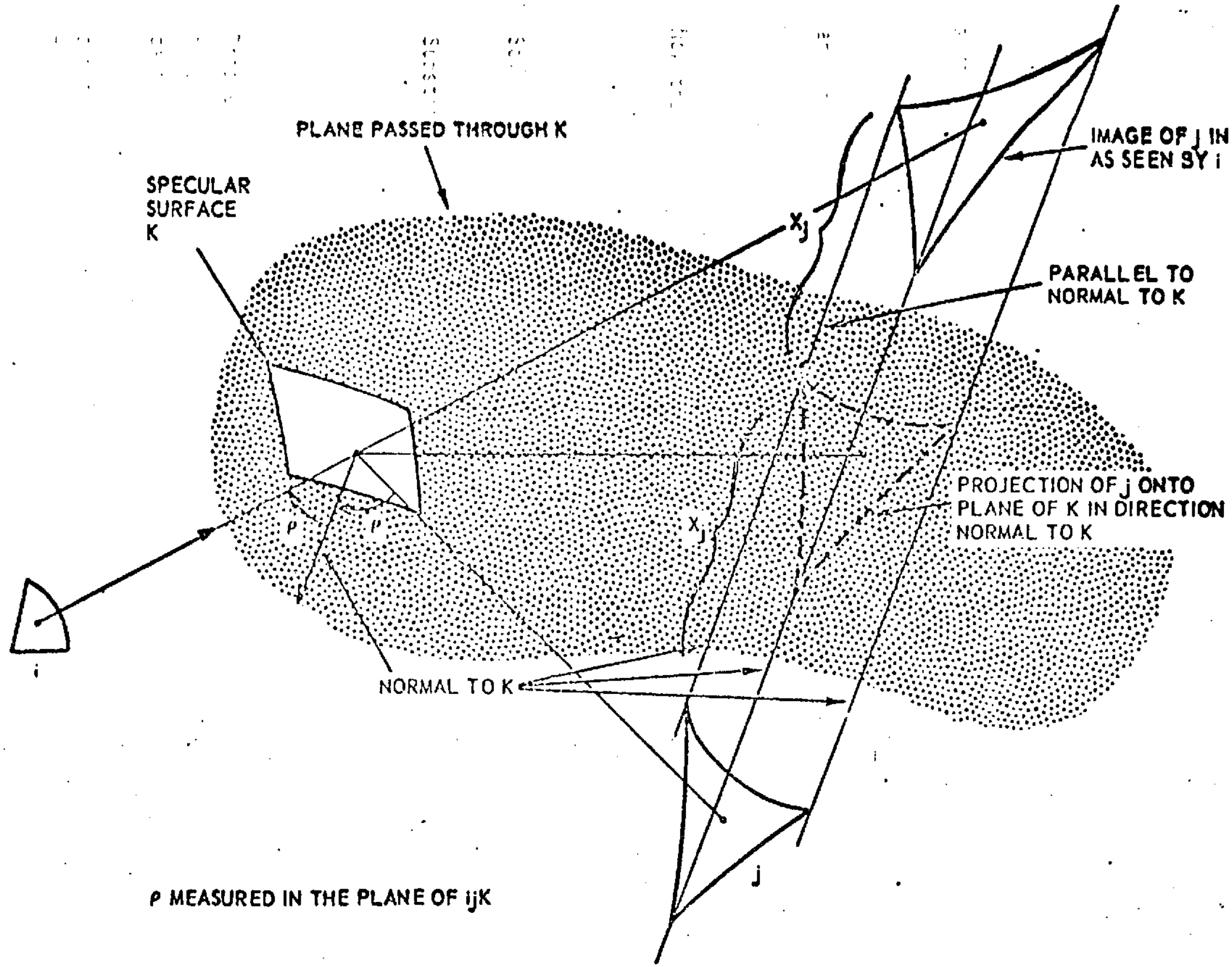


FIGURE 1 ILLUSTRATION OF METHOD USED TO DETERMINE SPECULAR SURFACE REFLECTED VIEW FACTORS

Here $F_{ij(k)}$ is the angle factor from i to j as seen in the specular surface k, $F_{ij(k,l)}$ is the angle factor from i to j as seen in the double specular reflection from k and l. There are an infinite number of possible combinations of these multi-reflections. It is evident that the interchange factors account for the specularly reflected radiant flux from the reflecting surface. This portion of total leaving flux is not a component of the radiosity of that surface. The radiosity may be written

$$B_i = \epsilon_i \sigma T_i^4 + \rho_i H_i, \tag{18}$$

and, for ns surfaces,

$$H_i = \frac{1}{A_i} \sum_{j=1}^{ns} B_j A_j E_{ji} \tag{19}$$

Now the interchange factors obey the reciprocity relation

$$A_i E_{ij} = A_j E_{ji} \tag{20}$$

So,

$$H_i = \sum_j B_j E_{ij} \tag{21}$$

Substitution into the equation for B results in

$$\sum_j (\delta_{ij} - \rho_i E_{ij}) B_j = \epsilon_i \sigma T_i^4 \tag{22}$$

This equation represents a set of linear, simultaneous, inhomogeneous algebraic equations for the unknowns (B_j). The symbol δ_{ij} is the Kronecker delta function which is 1 when $i = j$ and is 0 when $i \neq j$.

Note that the coefficients of B_j in equation (22) do not form a symetric coefficient matrix since the off diagonal terms contain $-\rho_i E_{ij}$. This equation can be made symetric by multiplying each equation by A_i/ρ_i.

This gives

$$\sum_j \left(\frac{\delta_{ij} A_i}{\rho_i} - E_{ij} A_i \right) B_i = \frac{\epsilon_i A_i}{\rho_i} \sigma T^4 \qquad i = 1, ns \tag{23}$$

Written in matrix form this equation is

$$E\,B = T \tag{24}$$

Where E is a symetric coefficient matrix. The solution is

$$B = E^{-1} T = [e_{ij}^{-1}]\, T \tag{25}$$

or

$$B_i = \sum_{J=1}^{ns} e_{ij}^{-1} \frac{\epsilon i\, A_j}{\rho_j} \sigma\, T_j^4 \tag{26}$$

The net heat transfer rate asorbed by surface i is given by

$$Q_i = A_i \epsilon_i \left[H_i - \sigma\, T_i^4 \right] \tag{27}$$

Where H_i is given from equation (18) as

$$H_i = \frac{1}{\rho_i} \left[B_i - \epsilon \sigma T_i^4 \right]$$

Substituting in for H_i gives

$$Q_i = A_i\, \epsilon_i \left\{ \frac{1}{\rho_i} [B_i - \epsilon_i\, \sigma\, T_i^4] - \sigma\, T_i^4 \right\}$$

$$= \frac{A_i\, \epsilon_i}{\rho_i} \left\{ B_i - [\rho_i + \epsilon_i]\, \sigma\, T_i^4 \right\} \tag{28}$$

Substituting in for B_i from equation (26) into equation (28) gives

$$Q_i = \frac{A_i \epsilon_i}{\rho_i} \left\{ \sum_{J=1}^{ns} \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_j} \sigma T_j^4 - [\rho_i + \epsilon_j] \sigma T_i^4 \right\}$$

$$= \frac{A_i \epsilon_i}{\rho_i} \left\{ \sum_{\substack{J=1 \\ J \neq i}}^{ns} \frac{e_{ij}^{-1} \epsilon_j A_j}{\rho_j} \sigma T_j^4 - \left[\rho_i + \epsilon_i - \frac{e_{ij}^{-1} \epsilon_i A_i}{\rho_i}\right] \sigma T_i^4 \right\} \qquad (29)$$

Since, in steady state, $Q_i = o$, and $T_i^4 = T_j^4$ for all i and j we can conclude that

$$\rho_i + \epsilon_i - \frac{e_{ij}^{-1} \epsilon_i A_i}{\rho_i} = \sum_{\substack{J=1 \\ J \neq i}}^{ns} e_{ij}^{-1} \frac{\epsilon_j A_j}{\rho_j}$$

Making the above substitution in equation (29) gives

$$Q_i = \sum_{J=1}^{ns} \sigma \frac{\epsilon_i \epsilon_j A_i A_j e_{ij}^{-1}}{\rho_i \rho_j} \left[T_j^4 - T_i^4\right] \qquad (30)$$

If we define $\mathcal{F}$ as

$$\mathcal{F}_{ij} = \frac{\epsilon_i \epsilon_j A_j e_{ij}^{-1}}{\rho_i \rho_j} \qquad i \neq j \qquad (31)$$

$$\mathcal{F}_{ij} = \frac{\varepsilon_i \varepsilon_j A_i}{\rho_i \rho_j} [e_{ij}^{-1} - \rho_i / A_i] \qquad i = j$$

Then

$$Q_i = \sum_{J=1}^{ns} \sigma \mathcal{F}_{ij} A_i \left[T_j^4 - T_i^4\right] \qquad (32)$$

This equation gives the heat flux between surfaces. However, each surface can contain several nodes. The heat absorbed by for each node is determined by:

$$Q_n = \frac{A_n}{A_i} \sum_{J=1}^{ns} \sigma \mathcal{F}_{ij} A_i \left[T_j^4 - T_n^4 \right] \qquad (33)$$

Where n = the node number on surface i

Prior to each iteration, the temperature of the surfaces are determined by

$$T_i^4 = \frac{\sum_{n=1}^{nn} A_n T_n^4}{\sum_{n=1}^{nn} A_n} = \frac{\sum_{n=1}^{nn} A_n T_n^4}{A_i} \qquad (34)$$

Where nn = the number of nodes on surface i

Since the heat transfer rate given by equation (33) depends on the node temperature, stability considerations must be taken into account. This is handled by storing the following relation into the array containing the sum of the conductors used for time increment calculation

$$CON_n = 4 \frac{A_n}{A_i} \sigma T_n^3 \sum_{J=1}^{nc} \mathcal{F} A_{ij} \qquad (35)$$

Subroutine RADIR makes the calculations necessary to obtain Qn given by equation (33) and CONn given by equation (35). The following is a summary of the calculations:

A. The following are performed the first time through RADIR:
   1. From the user input values of E_{ij}, A_i, and ρ_i, the E matrix given by equation (24) is formed. Only half of the symetric matrix is stored to save space.
   2. The E matrix is inverted in its own space to get E^{-1} with elements e_{ij}^{-1}
   3. The $\mathcal{F} A_{ij}$ values are determined from equation (31) and stored in the surface connections data.

B. The following calculations are performed on each temperature iterations:
   1. The temperature of each surface is calculated by equation (34).
   2. The heat absorbed for each node is determined using equation (33) and is added to the Q array.

The routine utilizes data used for obtaining $\mathcal{F}A_{ij}$ in step A as working space for step B, thus, maximizing space utilization.

2.1.6.2 Radiation From External Source

As with the internally generated radiation, the solar (or any other external source radiation) interchange factor is defined by

$$E^*_{ij} = F_{ij} + \sum_k \rho^{*s}_k F_{ij}(k) + \sum_k \sum_l \rho^{*s}_k \rho^{*s}_l F_{ij}(k,l) + \cdots$$

Where ρ^{*s}_k is the solar specular reflectance of surface K

$F_{ij}(K)$ is the angle factor from i to j as seen in the specular surface κ

$F_{ij}(K,l)$ is the angle factor from i to j as seen in a double specular reflection from j to l to k back to i

The interchange factors as defined above accounts for the specularly flux reflected from the surface. Thus, since the specular component of the flux is assumed to go directly from surface i to surface j by the interchange factor, Eij, this portion of the total flux is not a component of the radiosity for the intermmediate surfaces (k and l above). The radiosity of surface i is given by

$$B^*_i = \rho^*_i H^*_i \qquad (36)$$

Where B^*_i is the radiosity (energy leaving)

H^*_i is the incident energy

ρ^*_i is the diffuse reflectance

The energy incident upon a surface is given by

$$H_i = \sum_{J=1}^{ns} B^*_j E^*_{ij} + S_i \qquad (37)$$

Where S_i is the energy directly incident on surface i from an external source

Substituting equation (36) into (37), multiplying by A_i/ρ_i^* and simplifying gives the following relation for the radiosity

$$\left[\frac{A_i}{\rho_i^*} - E_{ii}^* A_i\right] B_i^* - \sum_{\substack{J=1 \\ J\neq i}}^{n} E_{ij}^* A_i B_j^* = S_i A_i \qquad i=1,n \tag{38}$$

This set of n equations can be written in matrix form as

$$E^* B^* = S \tag{39}$$

Note that the equations are written so that E* is a symetric matrix, which has the solution for B*

$$B^* = E^{*-1} S \quad \text{or} \quad B_i = \sum_{J=1}^{n} [e_{ij}^*]^{-1} S_j \, A_j \tag{40}$$

Where $[e_{ij}^*]^{-1}$ is the ijth element of the inverse of the E* matrix

The heat flux absorbed by the i th surface is given by

$$\frac{Q_i^*}{A_i} = \alpha H_i \tag{41}$$

But from equation (36)

$$H_i = \frac{B_i}{\rho_i^*} \tag{42}$$

Combining equations (40), (41), and (42) gives

$$Q_i^* = \sum_{J=1}^{n} e_{ij}^{*-1} \frac{\alpha_i}{\rho_i^*} A_j A_i S_j \tag{43}$$

If we define

$$\mathcal{F}_{ij}^* = e_{ij}^{*-1} \frac{\alpha_i}{\rho_i^*} A_j \tag{44}$$

Then the absorbed heat flux is given by

$$Q_i^* = \sum_{J=1}^{n} \mathcal{F}_{ij}^* A_i S_j \tag{45}$$

Equation (45) gives the heat absorbed by each surface. However, each surface may contain several temperature nodes. The absorbed heat for each node is given by:

$$Q_n^* = \frac{A_n}{A_i} Q_i^* \tag{46}$$

Where A_n is the area of the node

Subroutine RADSØL was written to make necessary calculations to obtain Q_n^* given by equation (46). The following is a summary of the calculations:

A. The following calculations are made the first time through RADSOL:

1. From the user input values of E^*_{ij}, ρ_i^*, and Ai, the E* matrix given by equation (39) is formed. Only one half is stored since E* is symetric.
2. The E* matrix is inverted in its own space to get E^{*-1} with elements, e_{ij}^{-1}.
3. The $\mathcal{F}^*_{ij} A_i$ values are determined from equation (44) and stored in the surface connections data.

B. The following calculations are performed on each temperature iteration:

1. The heat flux absorbed by each node is calculated by

$$\frac{Q_i^*}{A_i} = \frac{1}{A_i} \sum_{J=1}^{n} \mathcal{F}^*_{ij} A_i S_i$$

2. The net heat absorbed by this wavelength radiation is calculated for each temperature node on each surface by

$$Q_n^* = A_n \frac{Q_i^*}{A_i}$$

This quantity of absorbed heat is added to the Q array for node n.

Note that the user may specify subroutine RADSOL for an many bands of radiation from an external source as desired. A single call is required for each band.

2.2 Fluid Flow Analysis

Subroutine PFCS was written as a SINDA user subroutine to provide the ability to perform fluid pressure/flow analysis for flow of an incompressible fluid in tubes. The fluid flow analysis of PFCS is integrated with the thermal analysis capability so that the temperature dependence of properties is included in the pressure balances. PFCS is called from the VARIBLES 2 user logic block.

PFCS performs a pressure-flow balance on a general flow network including the following effects:

(1) Friction pressure drop

(2) Orifices and fitting type pressure losses

(3) Valves

(4) Pumps

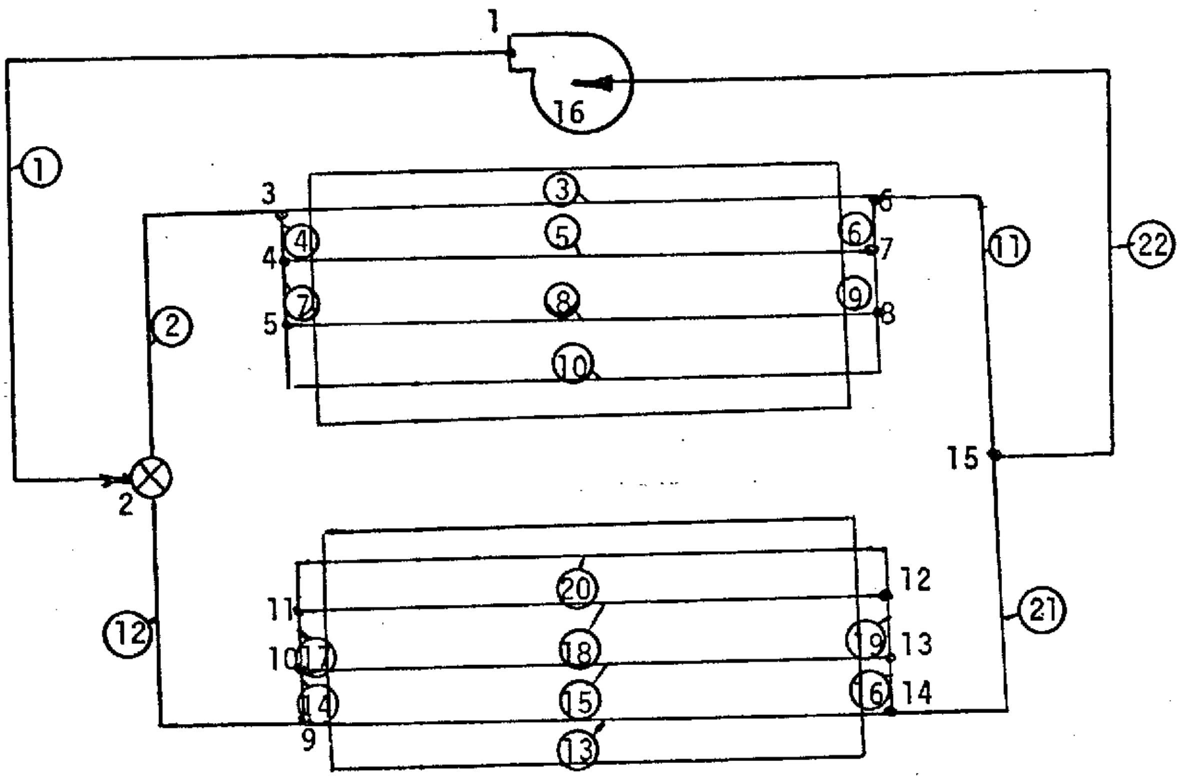
(5) Incoming flow sources at any pressure point in the system

The user describes the flow model to the subroutine by supplying the tube network connections and information concerning fluid properties, flow geometry, temperature model lumps, orifices, valves and pumps. Using this information, the subroutine determines the flow distribution required to satisfy (1) the conservation of mass at each node point and (2) equal pressure drops across tubes in parallel. The model used to describe the flow system and the analytical methods for determining the solution are described below.

2.2.1 Overall Flow Model Description

A flow problem may be analyzed with PFCS, simultaneously with a thermal analysis, so that the flow solution is continually updated based on the thermal conditions. To perform a flow analysis, the user must input a mathematical model of the flow system. The flow system is assumed to consist of a set of interconnected tubes such as the example shown in Figure 2 which consists of two radiator panels, each containing four tubes and connected so that they flow in parallel.

For clarity the following definitions are made at this point:



(X) Tube Numbers

XX Pressure Nodes

FIGURE 2 FLOW SYSTEM SCHEMATIC

(1) A tube is any single length of pipe between two pressure nodes. A tube "contains" fluid temperature nodes and may contain as many of these as required.

(2) A pressure node is located at each end of a tube. As many tubes as desired may be connected at a node junction and a node must exist at the junctions of two flow pipes.

We must make a mathematical model to describe the fluid flow information to the computer. The information required consists of:

(1) Identification of the pressure node numbers

(2) Identification of the tube numbers and the two pressure nodes connected by tube

(3) The fluid temperature nodes contained in each tube

(4) The flow geometry for each temperature fluid nodes

(5) The number of "head losses" for items such as orifices

(6) Fluid property information

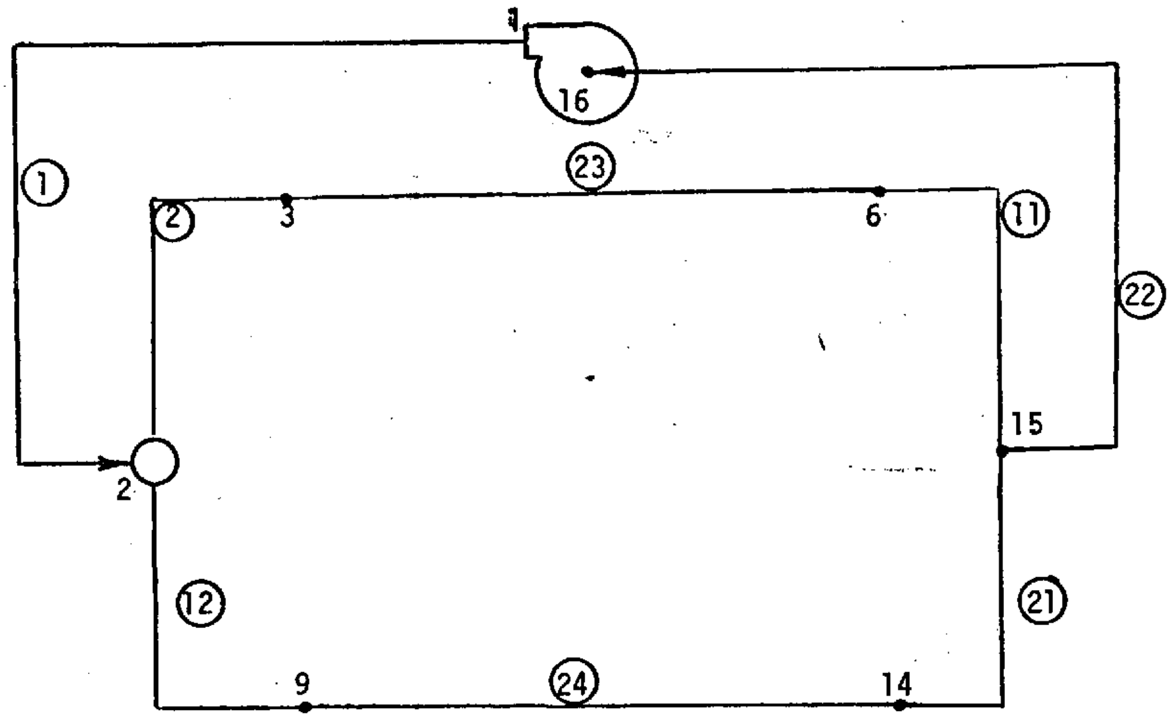
(7) Valve connections and characteristics

(8) Pump characteristics

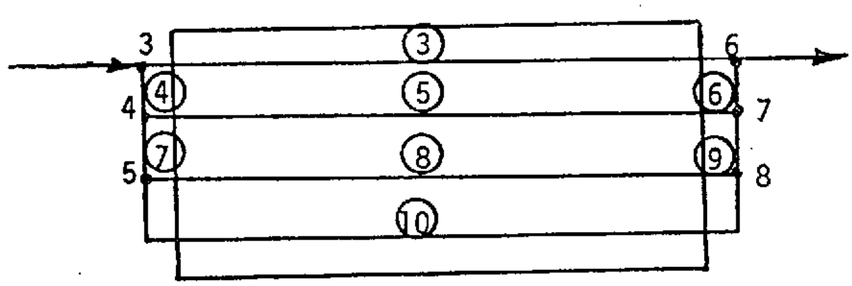
To build a flow mathematical model, a schematic of the flow system is needed. As shown in Figure 2, the pressure nodes and tubes may be superimposed on the schematic. It is also helpful to impose the fluid temperature lump numbers for each tube.

To facilitate speedy analysis on a general flow problem, provisions have been made for the user to divide the flow system network into subnetwork elements. For example, the flow system shown in Figure 2 could be divided as shown in Figure 3. Tubes 23 and 24 are added in the main network as shown in 3(a) to replace subnetwork elements 1 and 2. The subnetwork elements 1 and 2 which are shown in Figures 3(b) and 3(c) are then input as separate network elements. This type of subdivision allows the solution to be obtained by solving two sets of 6 simultaneous equations and one set of 8 equations rather than the original set of 16 simultaneous equations. This type of subdivision has been found to enhance the solution speed and accuracy for problems with a large number of nodes.

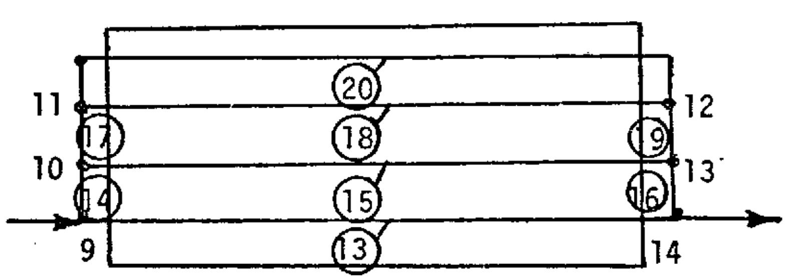
In summary, the pressure/flow solution is obtained by the following sequence:



a) Main Flow Network



b) Subnetwork No. 1



c) Subnetwork No. 2

FIGURE 3 MAIN NETWORK AND SUBNETWORKS

(1) The flow resistance is obtained for each fluid temperature lump in each tube including the effects of friction, orifices, and fitting type losses.

(2) The flow conductor valve is obtained for each tube by summing all the resistances of the fluid lumps in the tube, adding the valve and user supplied resistance to the sum, and inverting the resistance.

(3) A set of simultaneous equations is set-up and solved for each main system and subnetwork to obtain the pressures.

(4) The flow rates are then calculated.

A detail discussion of each element in the above sequence is described in the following subsections.

2.2.2 Tube Conductor Determination

The value of the flow conductor is determined for each tube by first calculating the flow resistance for each temperature fluid lump contained in the tube, summing these resistances up to obtain the flow resistance of the tube and inverting the tube resistance to get the conductance. Flow conductance is defined by the relationship

$$\dot{W}_{ij} = GF_{ij}[P_i - P_j] \tag{47}$$

Where W_{ij} = flow rate between pressure nodes i and j
GF_{ij} = flow conductance between nodes i and j
P_i = pressure at pressure node i
P_j = pressure at pressure node j

The flow resistance for each lump is then

$$R_K = \frac{1}{GF} = \frac{\Delta P_k}{\dot{W}}$$

Where R_k = flow resistance for lump k
ΔP_k = pressure drop for lump k

But ΔP_k is given by

$$\Delta P_k = \left(f_k \cdot ffc \cdot \frac{L_k}{D_k} + K \right) \frac{W^2}{2g_c \rho_k A^2} \tag{48}$$

Where f_k = the friction factor for lump k

ffc = the friction factor coefficient

L_k = the lump length for lump k

D = the lump hydraulic diameter for lump k

K = the dynamic head losses for lump k

$\dot{W}$ = the flow rate

g_c = the gravitational constant

ρ_k = the fluid density for lump k

A = the flow area

The flow resistance is then given by

$$R_k = \left(f_k \, ffc \, \frac{L_k}{D_k} + K \right) \frac{\dot{W}}{2 g_c \, \rho_k A^2} \tag{49}$$

Two options are available for obtaining the friction factor, f_k. These are (1) internal calculations for all flow regimes and (2) internal calculation for laminar flow and obtained from a table of f vs Re (where Re is the Reynold's number) for transition and turbulent flow. For the first option the internal calculations for the three flow regimes are:

<u>Laminar Regime</u>: $Re_k \leq 2000.$

$$f_k = \frac{64}{Re_k} \tag{50}$$

Where f_k = friction factor for lump k

Re_k = Reynolds number for lump k

<u>Transition Regime</u>: $2000 < Re_k < 4000$

$$f_k = .2086082052 - .1868265324 \left[\frac{Re_k}{1000}\right] + .06236703785 \left[\frac{Re_k}{1000}\right]^2 - .0065545818 \left[\frac{Re_k}{1000}\right]^3 \tag{51}$$

<u>Turbulent Regime</u>: $Re_k \geq 4000$

$$f_k = \frac{.316}{\left(Re_k\right)^{.25}} \tag{52}$$

Equation (51) for the transition regime is a curve fit between the laminar and turbulent regimes which was derived to match the two curves in a continuous manner. It is merely an arbitrary curve in this undefined region. A curve of the friction factor vs Reynold's number given by the above relations is shown in Figure 4.

The second option for friction factor uses equation (50) for the laminar regime and a user input curve of f_k ve Re for the other regimes. The options available for input of the dynamic head loss, $\mathcal{K}$, include (1) an input constant or (2) a tabulated curve of $\mathcal{K}$ vs Re.

To obtain the conductance for each tube, the flow resistances for all the lumps in the tube are added and then inverted, giving

$$GF_{ij} = \frac{1}{\sum_k R_k} \tag{53}$$

2.2.3 Valve Analysis

Provisions have been included in subroutine PFCS for valves to be included in the flow balance. The valve pressure drop is characterized by the following equation for each side of the valve;

$$\Delta P = E\left[\frac{\dot{\omega}}{\chi}\right]^2$$

where ΔP = valve pressure drop

E = valve pressure drop factor (user input)

$\dot{\omega}$ = flowrate through the side of the valve under consideration

χ = the fraction of the valve opening ($\chi = 0$ indicates valve closed; $\chi = 1.0$ indicates valve full open)

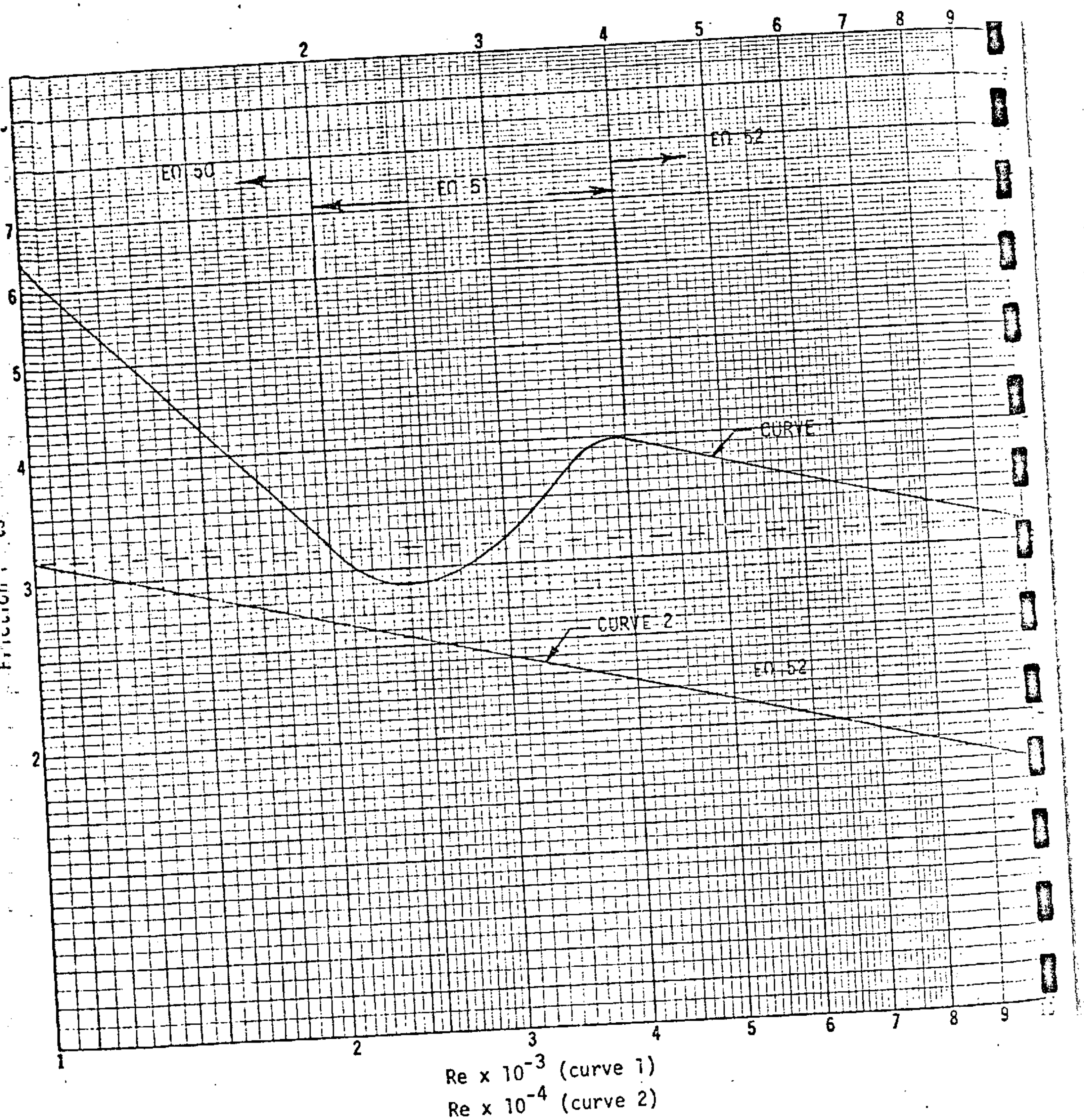


FIGURE 4 Friction Factor vs Reynolds Number

Three basic types of valves are available in PFCS which give different characteristics for the dynamics of the valve position χ. These types are:

(1) Rate Limited

(2) Polynomial

(3) Shut-off

A number of variations are available for each valve type. For instance, each of the above may be either one sided or two sided. If a valve is two sided, the valve position of side 2, χ_2, is related to that of side one by

$$\chi_2 = 1.0 - \chi_1$$

If the valve is one sided, either side one or side two may be used. Provisions are included for a valve time constant to be included with the polynomial valve.

The methods used to obtain the valve positions for each of the three methods are described below.

2.2.3.1 Rate Limited Valve

The valve position for the rate limited valve is obtained by an approximate integration of the valve rate of movement, $\dot{X}$. $\dot{X}$ depends on the temperature difference between the valve control set point temperature and the sensor temperature as shown in Figure 5. With this characteristic, the valve has no movement as long as the valve temperature error, ΔT, is within the dead band. Outside the dead band, the velocity of the valve increases linerarly as the error increases to a maximum rate, $\dot{X}$ max. The dead band, rate of velocity increase, $d\dot{X}/d(\Delta T)$, and the maximum velocity are controlled by user input.

The relations used to obtain the valve positions are as follows:

$$X^{i+1} = X^i + (\dot{X}^{i+1})\,(\Delta\tau) \qquad (54)$$

Where
X^{i+1} = valve position at iteration i+1
X^i = valve position at iteration i
$\dot{X}^{i+1}$ = valve velocity at iteration i+1
$\Delta\tau$ = the problem time increment

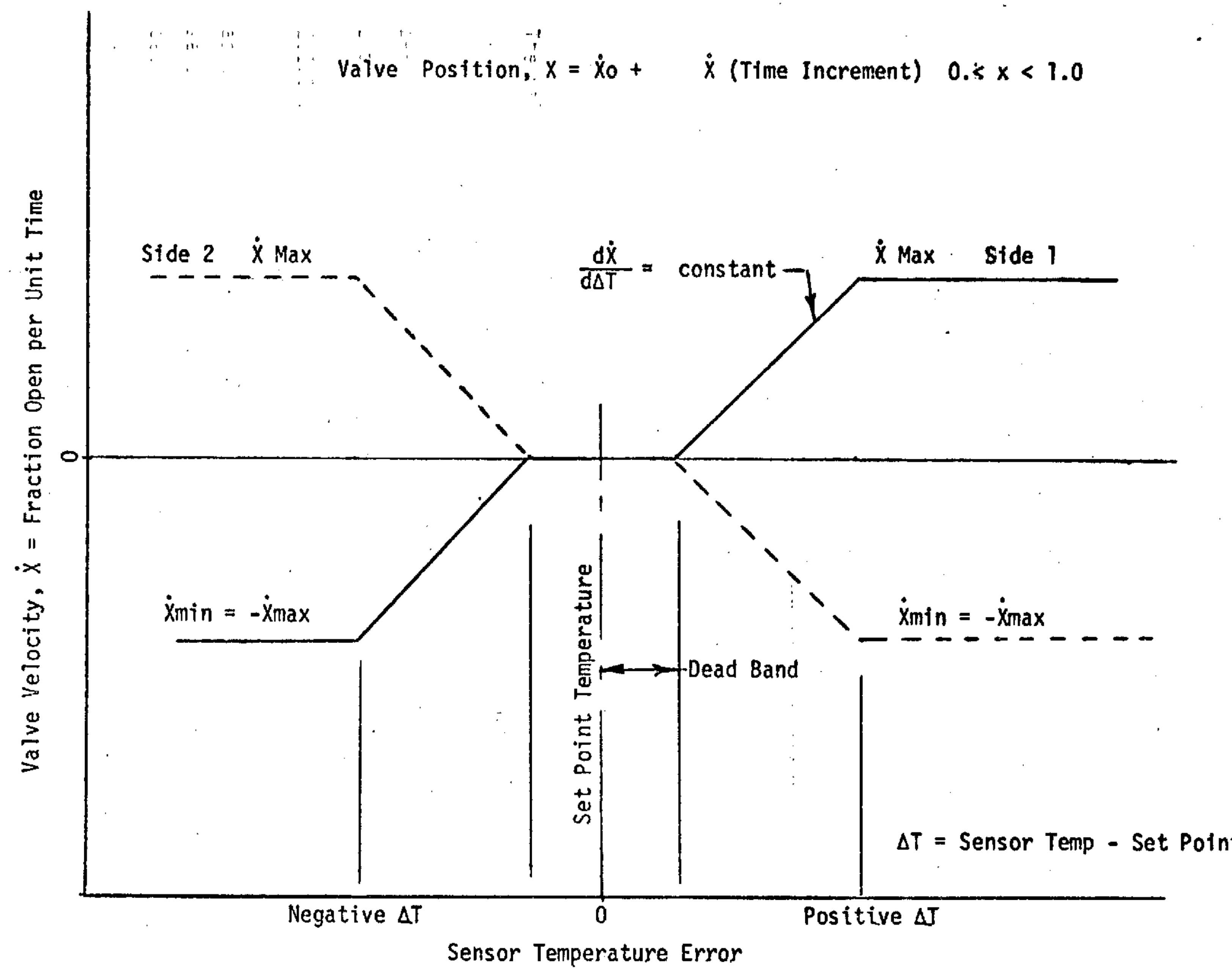


Figure 5 Rate Limited Valve Operation

The valve position is limited by

$$X\ min \leq X^{i+1} \leq X\ max$$

Where X min and X max are input limits on the valve position. The valve velocity, $\dot{X}^{x+1}$, in equation (54) is given by:

$$\dot{X}^{i+1} = 0 \text{ if } \left| Tsen - Tset \right| \leq Tdb$$

Where

$Tsen$ = Sensor lump temperature
$Tset$ = Set point temperature
Tdb = Valve dead band temperature

$$\dot{X}^{i+1} = \frac{d\dot{X}}{d(\Delta T)} \left[Tsen - Tset - Tdb \right] \text{ if } Tsen > Tset + Tdb$$

$$\dot{X}^{i+1} = \frac{d\dot{X}}{d(\Delta T)} \left[Tsen - Tset + Tdb \right] \text{ if } Tsen < Tset - Tdb$$

The valve velocity is limited by

$$\dot{X}min \leq \dot{X}^{i+1} \leq \dot{X}max$$

After the valve position for side 1 is obtained from equation (54), the side 2 position is obtained from $X_2 = 1.0 - X_1$

2.2.3.2 Polynomial Valve

The polynomial valve determines the steady state valve position as a 4th degree polynomial function of the temperature error between the sensor lump and the set point. A valve time constant is then applied to determine how far between the previous position and the new steady state position the valve will move. The steady state position, X_{ss}, is given by

$$X_{ss} = Ao + A_1 \Delta T + A_2 \Delta T^2 + A_3 \Delta T^3 + A_4 \Delta T^4$$

Where ΔT = Tsen - Tset
Tsen = the sensor lump temperature
Tset = the set point temperature
Ao, A_1, A_2, A_3, A_4 = input constants

The valve position, X^{i+1} is then determined by

$$X^{i+1} = X_{ss} + (X^{i} - X_{ss})\, e^{-\Delta\tau/\tau_c} \qquad (55)$$

Where X^{i+1} = valve positon at iteration i+1
X^{i} = valve position at iteration i
$\Delta\tau$ = problem time increment
τ_c = valve time constant

The valve position for side 2 is given by

$$X_2 = 1.0 - X_1$$

where X_1 is given by equation (55)
Note that this valve combines the capabilities of the polynomial valve and the proportioning valve described in Reference 6. If one desires to eliminate the effect of the time constant (and thus, give the valve an instantaneous response), a value for τ_c should be input which is small compared to the time increment, $\Delta\tau$. Also, either a constant value or a temperature lump number may be specified for the set point to permit use of the valve for proportioning between two sides.

2.2.3.3 Shut-off Valve

For side 1 of a shut-off valve the valve position decreases from X_{max} to X_{min} when the temperature of the sensor lump drops below the specified "off" temperature, T_{off}, and increased from X_{min} to X_{max} when the sensor lump exceeds a second specified temperature, T_{on}. T_{on} must be greater than T_{off}. Side 2 works in reverse of side 1. The valve position increased from X_{min} to X_{max} when the sensor temperature drops below the specified T_{on} and decreases from X_{max} to X_{min} when the sensor lump increases above the off temperature, T_{off}. For side 2, T_{off} must be greater than T_{on}. Note that, if the shut-off valve is a two sided valve with both sides active, the valve is a switching valve.

2.2.3.4 Valve Flow Resistance Calculations

The valve pressure drop on side one is assumed to be given by:

$$\Delta P = E \left[\frac{\dot{W}}{X} \right]^2 \tag{56}$$

Where E is an input constant

W is the flow through one side of the valve

X is the valve position (fraction of total possible distance)

Since flow resistance is $\Delta P/\dot{W}$, the valve flow resistance is given by

$$R_v = \frac{E\,\dot{W}}{X^2} \tag{57}$$

This value of flow resistance is calculated and added to the other flow resistances of the tube prior to performing the operation in equation (53) to find the value of the flow conductor for the tube.

Valves may be either one way or two way - i.e., be one tube or two tubes at the outlet. If only one tube exists on the valve outlet the flow resistance is calculated using equation (57) above. If a second tube exists, the resistance on side 2 is given by

$$R_{v2} = \frac{E_2\,\dot{W}_2}{(1 - X)^2} \tag{58}$$

2.2.4 Pressure-Flow Network Solution

As previously stated, the user may subdivide a system flow network into a main network and subnetwork elements. The elements which are subnetworks to the main network may also contain subnetwork elements but the subdivision can go no lower than two levels.

After the flow conductor values have been obtained by the methods described in Sections 2.2.2 and 2.2.3 a set of simultaneous equations are set up and solved for the main system and for each subnetwork. The subnetwork elements are all solved first and then, their equivalent flow conductor value is calculated. The value is inserted in the main system network and the system solution is obtained. The procedure is repeated until the problem is balanced.

A set of simultaneous equations are obtained by conservation of mass at each pressure node for each network and subnetwork. For any node i the conservation equation can be written as follows:

$$\sum W_{out} - \sum W_{in} = 0 \qquad (59)$$

Let $W_{in} = W_i$

and $\sum W_{out} = \sum_{J=1}^{nc} GF_{ij} [P_j - P_i]$

Then equation (59) becomes

$$\sum_{J=1}^{n} GF_{ij}[P_j - P_i] - W_i = 0 \qquad i=1,n \qquad (60)$$

Where GF_{ij} = flow conductor between pressure nodes i and j
P_i = pressure at node i
P_j = pressure at node j
W_i = flow rate added at node i
n = number of pressure nodes in the subnetwork

The above equation is a set of n simultaneous equations for P array. Pressure in the system or subsystem may be set at a specified level but the last (outlet) node must be specified. Equation (60) may be written in matrix form as:

$$GP = C \qquad (61)$$

Where

$$G = \begin{bmatrix} \sum GF_{ij} & -GF_{12} & -GF_{13} & \cdots\cdots \\ -GF_{21} & \sum GF_{2j} & -GF_{23} & \cdots\cdots \\ \vdots & \vdots & \vdots & \\ -GF_{n-1,1} & -GF_{n-1,2} & \cdots\cdots & \sum GF_{n-1,j} \end{bmatrix}$$

$$P = \begin{bmatrix} P_1 \\ P_2 \\ \cdot \\ \cdot \\ \cdot \\ P_{n-1} \end{bmatrix}$$

$$C = \begin{bmatrix} W_1 + GF_{1n} P_n \\ W_2 + GF_{2n} P_n \\ \cdot \\ \cdot \\ \cdot \\ W_{n-1} + GF_{n-1 1} \quad P_n \end{bmatrix}$$

P_n is the specified pressure. The above equations are solved for pressures at each point in the system and flow rates are then calculated for each tube by:

$$\dot{W}_{ij} = GF_{ij} (P_i - P_j) \tag{62}$$

Since the coefficient matrix given by equation (61) is symetric and positive definite the efficient square root or Symmetric Cholesky method was programmed to obtain the solution. This method is more accurate and faster than any other methods studied for this application.

Since the flow conductors are functions of the flow rate, the set of equations given by (61) are solved numerous times on each temperature iteration with a net set of GF_{ij} values for each solution. The iteration process continues until the change in the flowrates is within some user specified tolerance before proceeding to the next iteration.

2.2.5 Pump and System Pressure - Flow Matching

Concurrent with iterating the system flow equation to solution on each temperature iteration, the overall system pressure drop and flowrate must be matched to a pump characteristic. Several types of pump characteristics are available to the user as options. These are (1) system flow rate specified as a constant, (2) system flowrate specified as a known function of time, (3) pressure drop specified as a function of the flowrate in a tabulated form and (4) pressure drop specified as a function of flowrate with a fourth degree polynomial curve.

The first two options require no balancing of the pump with the system. Balancing is required for options (3) and (4) and iterative procedures have been devised to obtain the solution of the pump curve to the system characteristics with as few passes as possible through the system pressure/flow balancing loop for these options. The procedures used for these options are described below.

2.2.5.1 Tabulated Pump Curve Solution

The matching of a tabulated pump pressure rise/flow characteristic to the system pressure drop/flow characteristic is accomplished by the following procedure. See Figure 6 to aid in understanding the procedure.

Step 1 : The initial flowrate, W_1, at the system inlet is established either from user input on the first iteration or the system flow of the previous iteration for subsequent iterations.

Step 2 : Using $\dot{W}_1$, a solution to the flow network is obtained using the methods described in Sections 2.2.2, 2.2.3 and 2.2.4. Following this solution, ΔP_1 is available establishing point 1 on the true system characteristic curve shown in Figure 6.

Step 3 : Obtain an equation for the straight line approximation of the system characteristic (line 0, 1 for the first pass, line 1, 2 for the second pass, etc.)

$$\Delta P_S = C\ W_S + D$$

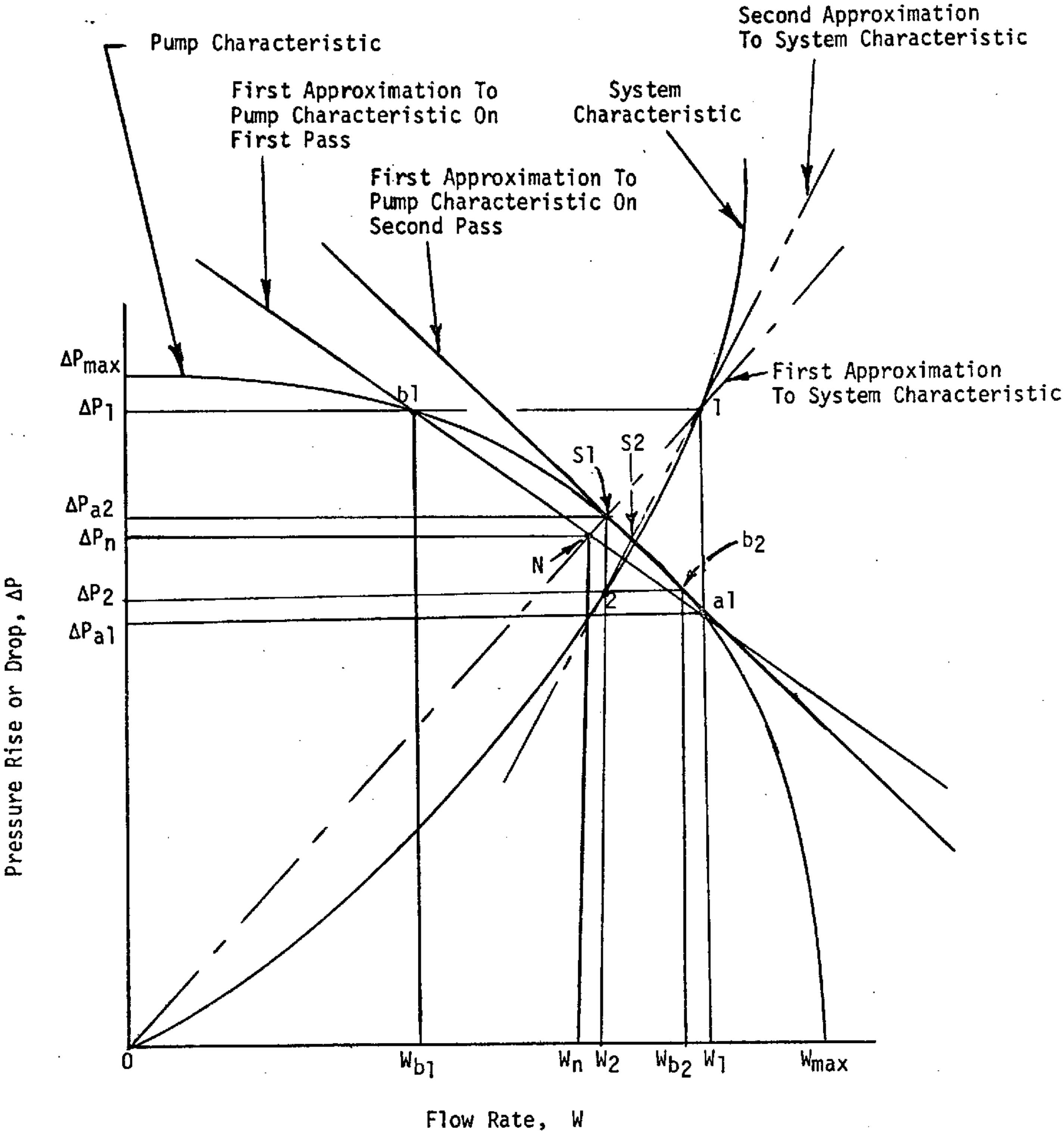


FIGURE 6 SYSTEM/PUMP CURVE SOLUTION

$$\text{where} \quad C = \frac{\Delta P_1 - \Delta P_0}{W_1 - W_0}$$

$$D = \Delta P_0 - \frac{\Delta P_1 - \Delta P_0}{W_1 - W_0} W_0$$

ΔP_s, W_s are the system pressure drop and flowrate values given by the approximate equation

ΔP_1, W_1 are the latest values for system pressure drop and corresponding system flowrate

ΔP_0, W_0 are the values for system pressure drop and corresponding system flowrate for the previous pass (These values are zero for the first pass)

Step 4 : Obtain the equation of the line connecting points a_1 and b_1 which is an approximation of the pump characteristic.

(1) Two points are determined on the pump characteristic curve:

(a) interpolate the tabulated characteristic at W_1 to obtain ΔP_{a1} (See Figure 6) to locate point a_1 at W_1, ΔP_{a1}. If W_1 is greater than W_{max}, set W_1 equal to W_{max} and ΔP_{a1} equal to zero.

(b) reverse interpolate the tabulated characteristic at ΔP_1 to obtain W_{b1} to locate point b_1 on the curve. If ΔP_1 is greater than ΔP_{max}, ΔP_1 is set to ΔP_{max} and W_{b1} is set to zero.

(2) Determine the coefficients A and B for the equation

$$\Delta P_p = A\dot{W}_p + B$$

$$\text{where} \quad A = \frac{\Delta P_1 - \Delta P_{a1}}{W_{b1} - W_1}$$

$$B = \Delta P_{a1} - \frac{\Delta P_1 - \Delta P_{a1}}{W_{b1} - W_1}$$

ΔP_p, W_p are the pump pressure rise and flowrate as given by the approximation.

Step 5 : Solve the approximate equations obtained in Steps 3 and 4 to obtain an approximate solution to the system characteristic and the pump characteristic (Point N) as follows:

$$W_N = \frac{D - B}{A - C}$$

$$\Delta P_N = AW_3 + B$$

Step 6 : Check the tolerance below where W_{n-1} is the previous W_N (W_1 for the first time through)

$$\text{Is} \quad \frac{W_N - W_{N-1}}{W_{N-1}} < .001$$

(1) If the above inequality equation is not satisfied repeat steps 4 through 6 substituting W_N for W_1 and ΔP_N for ΔP_1

(2) If the inequality is satisfied the point S1 (Figure 6) has been located. Continue with step 7. The final flowrate is W_2

Step 7 : Check the following tolerence

$$\text{Is} \quad \frac{W_2 - W_1}{W_1} < \text{TOL}^*$$

(1) If the above inequality equation is not satisfied, repeat steps 2 through 7 using the value of W_2 for W_1.

(2) If the inequality is satisfied, W_2 is the solution flowrate.

---

*TOL is the input pressure solution tolerance described on page 74

2.2.5.2 Polynomial Pump Curve Solution

When the user describes the pump curve with a polynomial curve fit, the pump characteristic is described by the relation

$$\Delta P_p = Ao + A_1W + A_2W^2 + A_3W^3 + A_4W^4$$

When this option is used, the procedure for matching the pump characteristic to the system characteristic is identical to that described in Section 2.2.5.1 for the tabulated pump characteristic except Steps 4 and 5 are replaced with the following:

Step 4 : Obtain the coefficients of the 4th order equation to be solved

Since:

$$\Delta P_p - \Delta P_s = 0$$

$\Delta P_s = C\ W_s + D$ (C and D are obtained from Step 3)

$$\Delta P_p = Ao + A_1W_p + A_2W_p^2 + A_3W_p^3 + A_4W_p^4$$

The solution occurs when

$$\Delta P_s = \Delta P_p$$

Then the equation for $\dot{W}_N$ is

$$(Ao-D) + (A_1-C)\ W_N + A_2W_N^2 + A_3W_N^3 + A_4W_N^4 = 0$$

Step 5 : Solve the equation for $\dot{W}_N$ using the Newton-Raphson Method of solution for a fourth order polynomial

The remaining steps are identical to that given in Section 2.2.5.1.

3.0 MODIFICATIONS TO SINDA SUBROUTINES

3.1 Preprocessor Modifications

Subroutine IMBED was written to convert actual conductor numbers, node numbers, array numbers or constant numbers which are input in the array data to their relative location in the G, T, A or K arrays respectively. The number to be converted in the array data is entered with an * followed by G, T, A, or K depending on the type of relative location desired. For instance, an array with the following input

12, *A10, *T5,*G101, END

would be changed so that the location of A10 in the A array would replace *A10 in the A12+1 location. The relative node number of actual node 5 would replace the *T5 and the relative conductor number of actual conductor number 101 would be placed in A12+3. The converted array might read

12, 102, 3, 21, END

Where A10 is located at location 102 in the A array, actual node 5 has relative number 3, and actual conductor 101 has relative conductor 21.

Subroutine IMBED is called from CODERD. Listings of IMBED, CODERD, and a modified overlay are supplied in Appendix A.

3.2 User Subroutine Modifications

A modification to the SINDA execution subroutine, CNFAST, was required to make it compatible with the radiant interchange subroutine, RADIR. This minor modification was required because the manner in which temperatures were calculated when the problem time increment is larger than the maximum convergence criterion for a given node was not compatible with the manner in which the convergence criteria information was carried over to CNFAST. A listing of the modified version of CNFAST is supplied in Appendix A.

A modification was made to user subroutine TPRINT to print temperatures in increasing order of actual node numbers. This modification increased the dynamic storage requirement to two locations for each node in the network.

4.0 USER SUBROUTINES

The capabilities described in Section 2.0 are available to the SINDA user through user subroutines which were added to the existing SINDA library[7]. This section presents a description and user input requirements for each subroutine. Table 1 summarizes the subroutines and the page that each user description is found.

The subroutine inputs rely heavily upon the capability to convert from actual array, constants, node, and conductor numbers to relative numbers in the array data. To use this capability the user may supply an actual array number, node number o conductor number by preceding the actual number with *A, *K, *T, or *G respectively. This causes the preprocessor to replace the entry with the relative number. Consider the example for array number 2 shown below.

2, *A14, *T5, *G7, END

In this example, following the preprocessor phase, *A14 will be replaced by the location in the A array of the Array No. 14 data, *T5 will be replaced by the relative node number for actual node No. 5, and *G7 will be replaced by the relative conductor number for actual conductor No. 7. This feature is used extensively for the input to user subroutines described below and is described in more detail in Section 3.1.

TABLE 1

NEW SINDA USER SUBROUTINES

SUBROUTINE NAME: CONV1

PURPOSE:

CONV1 calculates conductor values using the relations for convection heat transfer for flow in a tube. The relations used to obtain the film coefficient, h, are given by equations 2 thru 4 in section 2.1.1, depending on the flow regime (laminar, transitional or turbulent flow). h is then multiplied by the input area for heat transfer to obtain the conductor value. The conductor valve is stored in the input conductor location. Any number of conductors may be calculated with a single call. The flowrate array and the fluid properties data are addressed to tie the convection calculations with the pressure-flow solution. The first argument, AFLOW, is the first argument in the PFCS routine and identifies the flowrate array and fluid type data array needed by CONV1. APR is an array which references the specific heat, density, viscosity and thermal conductivity array. ADAT supplies other information needed.

RESTRICTIONS:

Should be called in the VARIABLES 1 block so that hA values are obtained every iteration.

CALLING SEQUENCE:

CONV1 (AFLOW, APR, ADAT)

where:

AFLOW is is first argument of the PFCS call and is of the following format: AFLOW(IC), AW, AP , AGF, AVP, AIFR, AFT, AFR, APD, END

AW array of flowrates per tube in the system

AP array of pressures per pressure node in the system

AGF array of pressure conductors per tube

AVP array of valve positions for all valves in the system

AIFR array of imposed flowrate per node

AFT array of fluid type data

AFR array of user added flow resistance per tube

APD array for output of pressure drops per tube

APR is the second argument of CONV1 and is also an array in the PFCS data input which contains fluid properties.
It is of the following format: APR(IC), CP, RO, MU, KT, GC, END

CP - is a doublet temperature dependent specific curve when input with the *A format

- is a constant specific heat valve if input as a real constant

RO - is a doublet temperature dependent density curve when supplied using *A

- is a constant density valve when supplied as a real constant

MU - is a doublet temperature dependent viscosity curve when supplied using *A

- is a constant viscosity valve when supplied as a real constant

KT - is a doublet temperature dependent thermal conductivity curve when supplied using *A

- is a constant conductivity valve when supplied as a real constant

GC - is the gravitational constant in the problem under consideration

ADAT - is an argument to CONV1 which contains convection information. It has the following format:

ADAT(IC), NG_1, AHT_1, $ITUBE_1$, NFL_1, $ITYPE_1$, X_1, $F1_1$, $F2_1$

⋮

NG_n, AHT_n, $ITUBE_n$, NFL_n, $ITYPE_n$, X_n, $F1_n$, $F2_n$

END

NG_i - is actual conductor number of the ith set of data

AHT - is the area for heat transfer

ITUBE is the tube number for obtaining flowrate

NFL - is the actual fluid lump number

ITYPE is the fluid lump type number

X - is the entry length

F1 - is the laminar fully developed coefficient

F2 - is the laminar entry length coefficient

SUBROUTINE NAME: CONV2

PURPOSE:

CONV2 calculates convection conductor values from a user input curve of Stanton number (St) vs Reynolds number (Re). The film heat transfer coefficient is obtained by (1) interpolating a curve of $St \cdot Pr^{2/3}$ versus Re to obtain $St \cdot Pr^{2/3}$ and (2) using the relation $h = k/D\ (St \cdot Pr^{2/3}) \cdot Re \cdot Pr^{1/3}$ to obtain h. h is multiplied by the heat transfer area to obtain the conductor values which is stored in the proper conductor location. The flow data used by subroutine PFCS is referenced by the arguments AFLOW and APR to obtain flowrate, type data and fluid property data thus tying the convection and flow analysis together.

RESTRICTIONS:

Should be called in the VARIABLES 1 block so that hA values are calculated on each iteration prior to the temperature calculation.

CALLING SEQUENCE:

CONV2(AFLOW, APR, ADAT)

where:

AFLOW - is the first argument of the PFCS call and is of the following format:
AFLOW(IC), AW, APR, AGF, AVR, AIFR, AFT, AFR, APD, END

AW - array of flowrates per tube in the system

APR - array of pressures per pressure node in the system

AGF - array of pressure conductors per tube

AVP - array of valve positions for all valves in the system

AIFR - array of imposed flowrate per node

AFT - array of fluid type data

AFR - array of user added flow resistance per tube

APD - array for output of pressure drops per tube

APR - is an array in the PFCS data input which contains fluid properties. It is of the following format:
APR(IC), CP, RO, MU, KT, GC, END

CP - is a doublet temperature dependent specific curve when input with the *A
- is a constant specific heat valve if input as a real constant

RO - is a doublet temperature dependent density curve when supplied using *A

- is a constant density value when supplied as a real constant

MU - is a doublet temperature dependent viscosity curve when supplied using *A

- is a constant viscosity valve when supplied as a real constant

KT - is a doublet temperature dependent thermal conductivity curve when supplied using *A

- is a constant conductivity value when supplied as a real constant

GC - is the gravitational constant in the problem under consideration

ADAT - is an argument to CONV1 which contains convection information. It has the following format:

```
ADAT(IC), NG1, AHT1, ITUBE1, NFL1, ITYPE1, AHST1
          .    .     .       .     .       .
          .    .     .       .     .       .
          .    .     .       .     .       .
          NGn, AHTn, ITUBEn, NFLn, ITYPEn, AHSTn
          END
```

NG_i - is conductor number of the ith set of data

AHT - is the area for heat transfer

ITUBE - is the tube number for obtaining flowrate

NFL - is the fluid lump number

ITYPE - is the fluid lump type number

AHST - is a doublet curve of $ST(PR)^{2/3}$ vs Re

SUBROUTINE NAME: CONV3

PURPOSE:

CONV3 calculates convection conductor values by interpolating a user supplied curve of heat transfer coefficient, h, versus tube flowrate. The conductor is then obtained by multiplying h times the area, A. A large number of conductors may be processed with a single call to CONV3. The flow data used by subroutine PFCS is referenced by the argument AFLOW to obtain flowrate and type data.

RESTRICTIONS:

CONV3 should be called from the VARIABLES 1 block to obtain updated hA values on each iteration.

CALLING SEQUENCE:

CONV3 (AFLOW, ADAT)

where:

AFLOW - is the first argument of the PFCS call and is of the following format:
AFLOW(IC), AW, APR, AGF, AVR, AIFR, AFT, AFR, APD, END

AW - array of flowrates per tube in the system

APR - array of pressures per pressure node in the system

AGF - array of pressure conductors per tube

AVP - array of valve positions for all valves in the system

AIFR - array of imposed flowrate per node

AFT - array of fluid type data

AFR - array of user added flow resistance per tube

APD - array for output of pressure drops per tube

ADAT - is an argument to CONV1 which contains convection information.
It has the following format:

$ADAT(IC), NG_1, AHT_1, ITUBE_1, AHW_1$

$\vdots \quad \vdots \quad \vdots \quad \vdots$

$NG_n, AHT_n, ITUBE_n, AHW_n, END$

NG_i - is conductor number of the ith set of data

AHT - is the area for heat transfer

ITUBE - is the tube number for obtaining flowrate

AHW - is a doublet array of heat transfer coefficient vs flowrate

SUBROUTINE NAME: FLOCN1 or FLOCN2

PURPOSE:

Subroutine FLOCN1 and FLOCN2 calculate thermal conductor values required for thermal characterization of fluid flowing down a tube. The conductor values are obtained by multiplying the tube flowrates times the specific heat for each of the conductors identified in the ADAT array. Both subroutines reference the flowrate array, AW, and FLOCN1 references the ACP array which gives the specific heat vs temperature relationship. FLOCN2 assumes a constant value for specific heat.

The conductor values referenced in the ADAT array must also be supplied in the CONDUCTOR DATA block as one-way conductors with the proper connections identified. Any dummy value may be supplied for the initial flow conductor values since these values will be replaced following the first call to FLOCN1 or FLOCN2. These subroutines are called from VARIABLES 1.

RESTRICTIONS:

Must be called from VARIABLES 1.

CALLING SEQUENCES:

FLOCN1 (AW, ACP, ADAT1) or FLOCN2 (AW, CP, ADAT2)

where

AW - is the array of flowrates per tube also referenced in subroutine PFCS

ACP - is a doublet array of specific heat versus temperature

Cp - is a constant value of specific heat

ADAT1 - is the array which identifies the conductor, the corresponding upstream lump and the tube number for each conductor. It is of the format:

ADAT1, NG_1, UPL_1, $ITUBE_1$
NG_2, UPL_2, $ITUBE_2$
⋮
NG_n, UPL_n, $ITUBE_n$, END

ADAT2 - is the array which identifies the conductor, number and the tube number for each conductor. It is of the format: ADAT2, NG1, ITUBE1, NG2, ITUBE2, - -, NGn, ITUBEn, END

NGi - is the ith conductor number (the *G notation is used)

UPLi - is the upstream lump number for conductor number NGi (The *T notation is used)

ITUBEi - is the tube number which contains the flow for the ith conductor $\dot{\omega} \cdot$ Cp product. For flow splitting or mixing junctions, ITUBEi should be the number of the connect tube containing the smallest amount of flow. For example, for a splitting junction the flow conductor which crosses the junction should contain the downstream tube. For a mixing junction ITUBE should be the upstream tube.

SUBROUTINE NAME: CONDT1 or CONDT2

PURPOSE:

Subroutines CONDT1 and CONDT2 calculate the value of conductance values as a product of a time variant argument W(t) and a temperature variant or constant argument Cp. The value of Cp is temperature dependent with CONDT1 and is constant for CONDT2. The subroutines were written primarily for the purpose of evaluating flow conductors for the case of flowrate a given function of time. However, they may be used for other time variant conductor applications.

RESTRICTIONS:

Should be called from VARIABLES 1.

CALLING SEQUENCE:

CONDT1 (ADAT1) or CONDT2 (ADAT2)

where ADAT1 is of the form

```
ADAT1, NG1, NLT1, ATIME1, ATEMP1
       NG2, NLT2, ATIME2, ATEMP2
        ,    ,      ,       ,
        ,    ,      ,       ,
        ,    ,      ,       ,
       NGn, NLTn, ATIMEn, ATMEPn, END
```

and ADAT2 is of the form:

```
ADAT2, NG1, ATIME1, CP1
       NG2, ATIME2, CP2
        ,     ,      ,
        ,     ,      ,
        ,     ,      ,
       NGn, ATIMEn, CPn, NED
```

The following definitions apply to the above.

NGi - the conductor number of the ith conductor addressed in ADAT1 or ADAT2. The *G notation should be used.

NLTi - the lump whose temperature will be used to interpolate the ATEMPi array to obtain the Cp constant. The *T notation should be used.

ATIMEi - the time variant array for determining the value of W(t) for the ith conductor. The *A notation should be used.

ATEMPi - the temperature variant array which is interpolated with the temperature of lump NLTi to obtain Cp in CONDT1. The conductor is calculated as the value of W(t)*Cp. The *A notation should be used.

Cp - the constant value which will be multiplied by W(t) from the ATIMEi array to obtain the conductor value for CONDT2.

SUBROUTINE NAME: RADIR

PURPOSE:

RADIR calculates the script-F values for infrared radiation heat transfer within an enclosure and uses these values to obtain the heat transfer during the problem. Several temperature nodes may be combined on a single surface for radiation heat transfer purposes. Also, the user may analyze problems with specular, diffuse or combinations of specular and diffuse radiation. See Section 2.1.6.1 for definitions and detailed description of methods.

RADIR calculates the script-F values on the initial call. This is performed by the procedure outlined in Section 2.1.6.1, Equations 23, 25 and 31. These values replace the EFT values in the SC array for future use. The heat flux values are then calculated on all iterations by:

(1) Calculating the temperature of each surface using equation 34

(2) Calculating the absorbed heat for each node by the relation of equation 33

The value given by equation 35 is added to the conductor sum for each node so that the proper convergence time increment may be obtained. As many enclosures as desired may be analyzed by each enclosure but each enclosure requires a different call to RADIR. RADIR must be called in VARIABLES 1.

RESTRICTIONS:

Must be called from VARIABLES 1 Surface nodes must be boundary nodes

CALLING SEQUENCE:

RADIR (A(IC)).

Where A is of the following format:

A(IC),SN,SE,SR,SC,NA,SP,END

SN,SE,SR,SC,NA, and SP are actual array numbers input using the *A procedure and are of the following formats

```
SN(IC),n,SN1,SA1,NN1,SN2,SA2,NN2,..............SNn,SAn,NNn,END
SE(IC),SE1,SE2-----SEn,END
SR(IC),SR1,SR2-----SRn,END
SC(IC),SNF1,SNT1,EFT1,SNF2,SNT2,EFT2,---SNFm,SNTm,EFTm,END
NA(IC),NNO(1,1),AN(1,1),NNO(1,2),AN(1,2)--NNO(1,NN1),AN(1,NN1)
       NNO(2,1),AN(2,1),NNO(2,2),AN(2,2)--NNO(2,NN2),AN(2,NN2)
          .        .        .        .          .         .
          .        .        .        .          .         .
          .        .        .        .          .         .

       NNO(n,1),AN(n,1),NNO(n,2),AN(n,2)--NNO(n,NNn),AN(n,NNn),END
SP(IC),SPACE,NSPACE,END
```

The following definitions apply in the above calling sequence:

A	Array idenitifcation for the array which identifies the other arrays containing the data
SN	Array number for the array containing surface numbers and areas
SE	Array number for the array containing the surface emissivities
SR	Array number for the array containing the surface reflectivities
SC	Array number for the array containing the surface connections data
NA	Array number for the array containing the temperature node numbers and areas
SP	Array number for the array containing the space which is used for obtaining script FA values and for subsequent temperature calculations
n	The number of surfaces
SN1,SN2,...SNn	Node number for surfaces - must be boundary nodes
SA1,SA2,....SAn	Total area for each surface
NN1,NN2,....NNn	Number of temperature nodes on each surface
SE1,SE2,....SEn	Emissivity values for each surface
SR1,SR2,....SRn	Diffuse reflectivity values for each surface
SNF1,SNT1,EFT1	Connections data: Surface number from, surface number to, E value from SNF1 to SNT1, etc.
NNO(X,Y)	Temperature node numbers on surfaces; Node number Y on surface X
AN(X,Y)	Area of node Y on surface X
NSPACE	Number of spaces needed to store script-FA values - NSPACE must be an integer values of n *n(n+1)/2
m	The number of surface connections

SUBROUTINE NAME: RADSOL

PURPOSE:

RADSOL calculates a pseudo script-F for radiation from an external source entering an enclosure and uses these values to calculate the net heat transfer to each node due to the entering source. A number of temperature nodes may be combined on a single surface for radiation purposes. Also, problems with specular, diffuse, or combinations of specular and diffuse radiation may be analyzed. Section 2.1.6.2 should be consulted for definitions and descriptions of methods.

RADSOL calculates the pseudo script-F values on the initial call. This is performed by equations 38, 40, and 44 of section 2.1.6.2. The values are stored in the EFT values of the SC array supplied by the user. The heat flux values are then calculated on each iteration by equations 45 and 46.

The user may analyze as many enclosures as desired by supplying a call statement for each enclosure. Also, a user may analyze several wave length bands by supplying a call to RADSOL for each wave length band.

RESTRICTIONS:

Must be called from VARIABLES I; Surface nodes must be boundray nodes

CALLING SEQUENCE:

RADSOL (A(IC))

Where the A array is of the following format:

A(IC),SN,SE,SR,HT,SC,NA,SP,END

SN,SE,SR,HT,SC,NA, and SP are actual array numbers input using the *A procedure and are of the following formats:

```
SN(IC), n,SN1,SA1,NN1,SN2,SA2,NN2,------Snn,SAn,NNn,END
SE(IC),SE1,SE2,------SEn,END
SR(IC),SR1,SR2,------SRn,END
HT(IC),SHT1,SHT2-----SHTn,END
SC(IC),SNF1,SNT1,EFT1,SNF2,SNT2,EFT2,---SNFm,SNTm,EFTm,END
NA(IC),NNO(1,1),AN(1,1),NNO(1,2),AN(1,2)---NNO(1,NN1),AN(1,NN1),
       NNO(2,1),AN(2,1),NNO(2,2),AN(2,2)---NNO(2,NN2),AN(2,NN2),
          '       '        '        '         '          '
          '       '        '        '         '          '
          '       '        '        '         '          '
       NNO(n,1),AN(n,1),NNO(n,2),AN(n,2)---NNO(n,NNn),AN(n,NNn),END
SP(IC),SPACE,NSPACE,END
```

The following definitions apply in the above calling sequence

A	Array identification for the array which identifies the other arrays containing the data
SN	Array number for the array containing surface numbers and areas
SE	Array number for the array containing the surface emissivities
SR	Array number for the array containing the surface reflectivities
HT	Array number for the array containing the incident heat curves or constant heat flux values
SC	Array number for the array containing the surface connections data
NA	Array number for the array containing the temperature node numbers and areas
SP	Array number for the array containing the space which is used for obtaining script values and for subsequent temperature calculations
SN1,SN2,...SNn	Node number for surfaces;must be boundary nodes
SA1,SA2,...SAn	Total area for each surface
NN1,NN2,...NNn	Number of temperature nodes on each surface
SE1,SE2,...SEn	Emissivity values for each surface
SR1,SR2,...SRn	Diffuse reflectivity values for each surface
SHT1,SHT2,...SHTn	Incident heat flow on surfaces; may identify curves containing incident values vs time
SNF1,SNT1,EFT1	Connections data: Surface number from surface number to, E value from SNF1 to SNT1, etc.
NNO(X,Y)	Temperature node numbers on surfaces: Node number Y on surface X
AN(X,Y)	Area of node Y on surface X
NSPACE	Number of spaces needed to store script-FA values - NSPACE must be an integer values of n/2(n+1)

SUBROUTINE NAME: HXEFF

PURPOSE:

This subroutine obtains the heat exchanger effectiveness either from a user constant or from a biariant curve of effectiveness versus the flow rates on the two sides. The effectiveness thus obtained is used with the supplied flow rates, inlet temperatures and fluid properties to calculate the outlet temperatures using the methods described in Section 2.1.3.4. The user may specify a constant effectiveness by supplying a real number or may reference and array number to specify the effectiveness as a bivariant function of the two flow rates. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXEFF should be called in the VARIABLES 1 block. The value for EFF, the first argument must never be zero. T_{out1} and T_{out2} must be boundary nodes.

CALLING SEQUENCE:

HXEFF(EFF,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2)

Where EFF - is (1) the effectiveness if real, (2)a curve number of a bivariant curve of effectiveness versus W1 and W2 if an array

W1,W2 - are the flow rates for side 1 and 2 respectively. May reference the flow rate array, AW+I where I is the tube number

CP1,CP2 - are the specific heat value for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties

TIN1,TIN2 - are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2

TOUT1,TOUT2 - are the outlet lump temperature locations sides 1 and 2 where the calculated values will be stored Must be boundary nodes

SUBROUTINE NAME: **HXCNT**

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relation described in Section 2.1.3.1 for a counter flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array number. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXCNT should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. T_{out1} and T_{out2} must be boundary nodes.

CALLING SEQUENCE:

HXCNT(UA,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2)

Where	UA	is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array
	W1,W2	are the flow rates for sideland side 2 respectively. May reference the flow rate array, AW+I where is the tube number
	CP1,CP2	are the specific heat values for side 1 and 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties
	TOUT1-TOUT2	are the outlet lump temperature locations (sides 1 and 2) where the calculated values will be stored Must be boundary nodes

SUBROUTINE NAME: HXCROS

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relations described in Section 2.1.3.3 for a cross flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array number. Any one of the following four types of cross flow exchangers may be analyzed (see Section 2.1.3.3 for the relations):

1) Both streams unmixed
2) Both streams mixed
3) Stream with smallest MCp product unmixed
4) Stream with largest MCp product unmixed

The type is specified by the last argument in the call statement. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for both sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXCROS should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. T_{out1} and T_{out2} must be boundary nodes.

CALLING SEQUENCE:

HXCROS(UA,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2,K)

Where	UA	is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array.
	W1,W2	are the flow rates for side 1 and side 2 respectively. May reference the flow rate array, AW+I where I is the tube number
	CP1,CP2	are the specific heat values for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperature dependent properties
	TIN1,TIN2	are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2
	TOUT1,TOUT2	are the outlet lump temperature locations(sides 1 & 2) where the calculated values will be stored Must be boundary nodes

K is the code specifying type of cross flow exchanger:

Both streams unmixed: K=1
Both streams mixed: K=2
Stream with small WCp Unmixed: K=3
Stream with large WCp Unmixed: K=4

SUBROUTINE NAME: HXPAR

PURPOSE:

This subroutine calculates the heat exchanger effectiveness using the relations described in Section 2.1.3.2 for a parallel flow type exchanger. The value of UA used in the calculations may be specified as a constant by supplying a real number or it may be specified as a bivariant function of the two flow rates by referencing an array. The user also supplies flow rates, specific heat values, inlet temperatures and a location for the outlet temperatures for each of the two sides. The flow rate array may be referenced to obtain flow rates and the temperature array may be used for temperatures. The specific heat values may be supplied as a temperature dependent curve or a constant value may be supplied.

RESTRICTIONS:

HXPAR should be called in the VARIABLES 1 block. The value for UA, the first argument must never be zero. T_{out1} and T_{out2} must be boundary temperatures.

CALLING SEQUENCE:

HXPAR(UA,W1,W2,CP1,CP2,TIN1,TIN2,TOUT1,TOUT2)

Where	UA	is (1) the heat exchanger conductance if real, (2) a curve number of a bivariant curve of conductance versus W1 and W2 if an array.
	W1,W2	are the flow rates for side 1 and 2 respectively. May reference the flow rate array,AW+I where I is the tube number
	CP1,CP2	are the specific heat values for side 1 and side 2 fluid respectively. Constant values may be input or arrays may be used for temperatures dependent curves.
	TIN1,TIN2	are inlet lump temperatures - Usually T(IN1) and T(IN2) where IN1 and IN2 are the inlet lumps on side 1 and side 2
	TOUT1,TOUT2	are the outlet lump temperature locations (sides 1 and 2) where the calculated values will be stored (should be boundary temperatures)

SUBROUTINE NAME: HEATER

PURPOSE:

This subroutine simulates an electrical heater with a control system which turns the heater on when the sensor lump temperature falls below the "heater on" temperature TON, and turns the heater off when the sensor lump rises above the heater off temperature, TOFF. When the heater is on, the input Q value is added to the Q location specified by the user. When the heater is off, no heat is added.

RESTRICTIONS:

HEATER must be called in the VARIABLES 1 block.

CALLING SEQUENCE:

HEATER(Q,QHT,KODE,TSEN,TØN,TOFF)

Wheren	TSEN	is the sensed temperature
	TON	is the heater on temperature
	TOFF	is the heater off temperature
	QHT	is the heater heat rate
	Q	is the location for storing the heat

KODE is an integer variable set by HEATER
= 1 if the heater was "on" at last call
= 0 if the heater was "off" at the last call
(User sets KODE for first call)

SUBROUTINE NAME: CABIN

PURPOSE:

This subroutine performs a thermal and mass balance on a cabin air system. The cabin air is assumed to be a two component gas mixture with one condensible component and one noncondensible component. The cabin air is assumed to be well mixed so that the temperature and specific humidity are constant throughout. The cabin may contain any number of entering streams each with different temperature and humidity conditions. The cabin air may transfer heat to any number of nodes in its surroundings with the heat transfer coefficient obtained by one of the the three options:

1. User input coefficient
2. Relations for flow over a flat plot
3. Relations for flow over a tube bundle

The relations describing the second and third options are given in Section 2.1.5. The mass transfer coefficient for determining the rate of condensation or evaporation is determined by the Lewis relation which relates the mass transfer coefficient directly to the convection heat transfer coefficient. By the Lewis Relation, if the diffusion coefficient is approximately equal to the thermal diffusivity, the Sherwood number is approximately equal to the Nusselt number, thus giving a direct relation. (See Section 2.1.5 for details). Mass and heat transfer rates are determined at each node that interfaces the cabin gas as well at entering and exiting streams and a new cabin gas temperature and humidity is determined each iteration based upon the heat and mass balance. An account is kept of the condensate on the walls when condensation occurs but the condensate is assumed to remain stationary and not flow to other wall nodes.

Limits are applied when necessary to prevent more condensation than the vapor existing under severe transient condition and to prevent evaporation of more liquid than exists at each wall lump.

As many cabins as desired may be analyzed in a given problem, but each must contain separate input information.

RESTRICTIONS:

CABIN must be called in VARIABLES 1.

CALLING SEQUENCE:

CABIN(A(IC) TC, TC, K1, K2)

The following definitions apply to the above calling sequence:

A	is an array containing arrays numbers which contain cabin input information

TC — The cabin gas temperature which must be a boundary node

K1,K2 — Storage locations needed by CABIN

The A array has the following format where the *A procedure is used:

A(IC),IF,PR,CN,H,FP,TB,SP,END

Where IF — Identifies an array containing the entering flow rate information. The format of the array is:

$IF(IC),NS,FR_1,PSI,TE_1,FR_2,PSI_2,TE_2{-}{-}{-}{-}FR_{ns},PSI_{ns},TE_{ns}$

PR — Identifies an array identifying array numbers for property values. The format of the array is:

PR(IC),NFLC,NMUO,NMUV,NCPO,NCPV,NKO,NKV,NLAT

CN — Identifes an array containing pertinent constants. The format of the array is:

CN(IC),RA,RV,VC,PC,XC,WV,PSIC,PO,TO,CONV

H — Identifies an array containing node numbers and convection heat transfer coefficient values for nodes surrounding the cabin gas. The format of the array is:

$H(IC),LN_1,\ HA_1,\ LN_2,\ HA_2,\ -\ -\ -\ LN_{n1},\ HA_{n1}$

FP — Identifies an array containing node numbers and information to permit calculation of convection coefficients for flat plates. The format is:

$FP(IC),LN_1,XX_1,XI_1,AI_1,VIWØ_1,LN_2,XX_2,XI_2,AI_2,$

$VINØ_2,{-}{-}{-}{-}{-}{-}{-}LN_{n2},XX_{n2},XI_{n2},AI_{n2},VIWØ_{n2}$

TB — Identifies an array containing node numbers and information to permit calculation of convection coefficients for tube bundles. The format is:

$TB(IC),LN_1,DI_1,AI_1,VIWØ_1,LN_2,DI_2,AI_2,VIWØ_2,{-}{-}{-}{-}{-}{-}LN_{n3},$

$DI_{n3},AI_{n3},VIWO_{n3}$

SP — Identifies an array which contains working space equal to or greater than three times the sum of the number of nodes with input heat transfer coefficients plus the number using flat plot relations plus the number using tube bundles.

The following symbol definitions apply in the above:

NS	Number of incoming streams
FR_i	Entering flow rate for stream i
PSI_i	Specific humidity for entering stream i
TE_i	Temperature of entering stream i
NFLC	Curve number for circulation flow rate vs time
NMUO	Curve number for noncondensible viscosity vs temperature
NMUV	Curve number for condensible viscosity vs temperature
NCPO	Curve number for noncondensible specific heat vs temperature
NCPV	Curve number for condensible specific heat vs temperature
NKO	Curve number for noncondensible thermal conduction vs temperature
NKV	Curve number for condensible thermal conduction vs temperature
NLAT	Curve number for latent heat of condensible vs temperature
RA	Gas constant for non-condensible component
RV	Gas constant for condensible component
VC	Cabin volume
PC	Cabin Pressure
XC	Molecular weight ratio, Mv/Mo
WV	Initial vapor weight in cabin
PSIC	Initial specific humidity for cabin
LN_i	Cabin wall lump
HA	Heat transfer coefficient times area
n1	Number of wall lumps which have input HA values
n2	Number of wall lumps which have HA calculated by flat plate relations
n3	Number of wall lumps which have HA calculated by tube bundle relations

xx_i	Distance from leading edge for flat plate heating for ith flat plate node
XI_i	Length of flat plate in flow direction for ith flat plate node
AI_i	Heat transfer area for flat plate or tube node
DI_i	Tube outside diameter for tubes in the bundle for ith tube node
VIWO	Ratio of velocity at the lump to the circulation flow rate
To	The reference temperature to be used for estimating the saturation pressure of the condensible component. Should be near the range of saturation temperature expected
Po	The saturation pressure at To for the condensible component
CONV	Conversion factor to make the quantity XLAM/Rv/To dimensionless where XLAM is the latent heat of vaporization and Rv is the gas constant for the vapor. If XLAM is BTU/lb, Rv is FT-LB/°R and To is °R, CONV=778.

SUBROUTINE NAME: PFCS

PURPOSE:

Subroutine PFCS determines the flow distribution in a set of general parallel/series fluid flow tubes so that the pressure drop values between any parallel flow paths are equal and flow is conserved. The following effects are included in the pressure drop calculations:

(1) pipe flow friction

(2) orifices and fittings

(3) valves

The effect of temperature dependent properties are included in the calculations. The properties are evaluated at the temperature of each fluid lump in each tube in evaluating the flow resistance when setting up the equations to be solved. A balance is made between the flow/pressure drop characteristics of the system and the flow/pressure rise of a pump for each system concurrent with the system pressure flow solution to obtain the incomming system flowrate. A detailed discussion of the equations and techniques used are described in Section 2.2. General flow charts of PFCS and supporting subroutines are shown in Fig. 7,8, & 9.

RESTRICTIONS:

Must be called from VARIABLES 2. The system of units used for the thermal and flow problems should be consistent.

CALLING SEQUENCE: PFCS (AFLOW, ADAT, NAME)

where AFLOW - is an array which references other arrays for flowrates, pressures, flow conductors, valve positions, imposed flowrates, fluid type data, user added flow resistances and pressure drops. It is of the following format where the *A conversion feature described in Section 3.1 is used to reference arrays.
AFLOW (IC), AW, APN, AGF, AVP, AIFR, AFT, AFR, APD, END

ADAT - is an array which identifies other arrays containing fluid property values, parameters needed for the pressure/flow solution numerical technique, the flow system network, valve data, pump data and a check outprint code. The integer count should be addressed. The format of ADAT is as follows where the *A format is used to address array values: ADAT(IC), APR, ASOL, ANET, AVLS, AP, KOP, END

NAME - is an array containing the name of the network (it may also be supplied as a Hollerith using the H format). Nine 6 character words should be used.

AFLOW ARGUMENTS

AW - is the array number of an array containing flowrates per tube for each system. The integer count must be addressed and it must contain the number of spaces exactly equal to the number of tubes in the system.

APN - is an array number for an array containing the pressures for each pressure node in the system. On input the user need only set up the space. The interger count must be addressed and it must contain the number of spaces exactly equal to the number of pressure nodes in the system.

AGF - is an array number for an array containing the flow conductors for each tube in the system. The user needs only to setup the space on input which must be exactly equal to the number of tubes in the flow system.

AVP - is an array number for an array containing valve positions in order of valve numbers. The interger count must be addressed and the number of input valves must be exactly equal to the number of valves. The user supplies the initial valve positions in this array.

AIFR - is the array location of an array of imposed flow sources for each pressure node. The interger count must be addressed and the array must contain the number of spaces exactly equal to the number of pressure nodes in the flow system.

AFT - is the array location of an array which contains fluid lump type data. The AFT array is of the following format:

AFT, WP_1, CSA_1, FLL_1, MFF_1, NHL_1, FFC_1

⋮

WP_{nt}, CSA_{nt}, FLL_{nt}, MFF_{nt}, NHL_{nt}, FFC_{nt}

END

nt - is the number of types

WP_i - is the wetted perimeter of fluid type i

CSA_i - is the cross sectional area for fluid type i

FLL_i - is the fluid lump length for fluid type i

MFF_i - is the curve of friction factor vs Reynolds number for Reynolds number greater than 2000 when greater than 0.
- is a key to use internal calculations methods for friction factor when MFF = 0.

NHL_i - is the number of head losses for type i when real
- is the number of an array of head losses vs Reynolds number when an integer

FFC_i - is a user input constant to be multiplied times the friction factor to modify it for type i.

AFR - is an array number of an array containing user added flow resistances for the tubes. This can be used to include the effects of changes in flow altitudes or the effects of valve types not available in the valve package or other known flow resistances. The integer count must be addressed and the number of values in the array must be exactly equal to the number of tubes in the flow system.

APD - is an array number of an array which will contain pressure drop values for all tubes in the flow system following a call to PFCS. The array is strictly for output purposes. The integer count must be addressed and the number of array valves must be exactly equal to the number of tubes in the system.

ADAT ARGUMENTS

APR - is an array identifying the fluid properties data and GC, the gravitational constant. It is of the following format: APR, CP, RO, MU, KT, GC, END

CP, RO, MU, KT, are the values of fluid specific heat, density, viscosity, and thermal conductivity respectively or the appropriate array reference (using the *A format). The value is constant for any of the properties if a real number is supplied. The integer count must be referenced when variable properties are used.

GC - is the gravitational constant. Table 2 gives the value for various system of units.

ASOL - is an array number of an array containing various numerical solution parameters needed by PFCS. ASOL is of the following format:

ASOL (IC), TOL, MXPASS, EPS, FRDF, END

TOL - is the solution tolerence on rate of change of flowrates from one pass to the next. The fraction of change must be within TOL for all tubes in any system or subsystem before a solution is reached. TOL must be greater than 0. A typical value is 0.001

MXPASS - is the maximum number of passes permitted in the balancing loop of PFCS to obtain a pressure-flow solution on any given iteration. This value should always be greater than 20 with a typical value of 100.

EPS - Not used but a space must be supplied.

FRDF - is the flowrate damping factor used to accelerate the rate of convergence for the iterative solution to the set of non-linear equations. This value should generally be between 0.5 and 1.0. Values of 0.5 to 0.7 have been found best for most turbulent flow problems.

ANET - is the array number of an array which identifies the tube connections, pressure nodes connected and fluid lumps contained in the tube. The format of ANET is as follows were the *A format is used for AD, APNPS, and AVL:

ANET,NNAME, APNPS, AVL
TUBE1, NFRM1, NTO1, KD1, AD1
TUBE2, NFRM2, NTO2, KD2, AD2
⋮
$TUBE_n$, $NFRM_n$, NTO_n, KD_n, AD_n, END

NNAME - is a six character name to be used for identifying the network in output statements, etc.

APNPS - is an array number (referenced using *A) of an array identifying nodes with specified pressures and is of the form

APNPS, NSPR1, NSPR2, - - - - - - NSPRn, END

NSPRi - is the ith pressure node with specified value

TUBEi - is the tube number of the ith connection

NFRMi - is the "from" pressure node for the ith connection

NTOi - is the "to" pressure node for the ith connection

KDi - is an integer code to identify the type of conductor for the ith connection. (See ADi below)

ADi - is the data to be used for calculating the conductor value for the ith connection.

If KDi < 0, the conductor value is the equivalent conductance of a subnetwork described by array ADi. ADi is then of identical format to ANET.

If KDi > 0, the conductor is obtained by the normal pressure drop equations and array ADi fluid lumps, fluid lump types and tube lumps that are contained in the tube. The form of ADi is

```
ADi, NFLMP1, ITYPE1, NTLMP1
     NFLMP2, ITYPE2, NTLMP2
       :        :        :
       :        :        :
       :        :        :
     NFLMPn, ITYPEn, NTLMPn, END
```

Where NFLMPi is ith temperature lump contained in the tube, ITYPEi is the NFLMPi fluid lump type, and NTLMP is the tube lump containing NFLMPi.

If $KD_i = 0$, the conductor calculation is not made, allowing the user to supply the pressure conductance value. AD_i is not used.

AVLS - is the array location (identified in array ADAT using the *A format) of an array which identifies the array location of the valve data for all the valves in the system.

AVL - is the array location (identified in array ANET using the *A format) of an array which identifies the array locations of the valve data for the valves in the network or subnetwork described by ANET.

AVLS and AVL are of the following format:

AVLS or AVL, AVLV1, AVLV2, - - - - AVLVNV, END

Where AVLVi is the array number (using *A) of an array which contains the valve data for the ith valve in AVLS or AVL. The format for the valve data arrays, AVLVi, is one of three forms depending on the valve type (rate limited, polynomial, or switching).

The format for a rate limited valve is:

AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, TSEN1, TSEN2, DB, RF, RL, END

The format for a polynomial valve is:

AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, TSEN1, TSEN2 A0, A1, A2, A3, A4, A5, VTC, END

The format for a switching valve is:

AVLV, NV, NTS1, NTS2, MODE, XMIN1, XMAX1, E, NSEN, T1, T2, END

The following definitions apply for the above arrays:

NV - Valve number

NTS1 - Tube number connected to side 1 of the valve

NTS2 - Tube number connected to side 2 of the valve

MODE - Operating mode: 1 - operating; 0 - not operating

XMIN1 - Side 1 minimum position; side 2 maximum position is (1.0 - XMIN1)

XMAX1 - Side 1 maximum position; side 2 minimum position is (1.0 - XMAX1)

E - The valve geometric factor relating pressure drop through the valve by

$$\Delta P = E(\text{flowrate/valve position})^2$$

TSEN1 - Sensor lump for side 1 or set point for side 2; If TSEN1 is an integer, it identifies the side 1 sensor lump to be controlled to (a) the set point for side 1 or (b) the sensor lump for side 2 (TSEN2). If the variable is input as a real number it represents a set point to which the side 2 sensor lump will be controlled.

TSEN2 - Sensor lump for side 2 or set point for side 1; If TSEN2 is an integer, it identifies the side 2 sensor lump to be controlled to (a) the set point for side 2 or (b) the sensor lump for side 1 (TSEN1). If the variable is input as a real number it represents a set point to which the side 1 sensor lump will be controlled.

A0, A1, A2, A3, A4, A5 - Polynomial curve fit coefficients for a curve fit of the steady state valve position vs sensed temperature error for side 1:

$$X1SS = A0 + A1 \cdot \Delta T + A2 \cdot \Delta T^2 + A3 \cdot \Delta T^3 + A4 \cdot \Delta T^4 + A5 \cdot \Delta T^5$$

DB - Dead band for the rate limited valve, degrees of temperature (See Figure 5).

RF - Rate factor, the rate of change of valve velocity to sensed temperature error ($d\dot{x}/d(\Delta T)$) as shown on Figure 5.

RL - Rate limit, the maximum valve velocity, $\dot{X}max$ (See Figure 5).

VTC - Valve time constant as described in Section 2.2.3.2. If a valve is desired with no time lag, a time constant which is very small compared to the problem time increment should be input. (VTC must be greated than zero).

NSEN - Sensor lump for switching valve

T1 - Side 1 off temperature or side 2 on temperature for switching valve

T2 - Side 2 off temperature or side 1 on temperature for switching valve

AP - is the array number of an array containing the pump data for the system specified in the ADAT array using the *A nomenclature. The format of the AP array is different for different types of pumps. If flowrate is a function of time the format is (where AW is supplied using *A):

AP, NPI, AW, END

If the flowrate is obtained using a tabulated pump curve the format is: (where ADP is supplied with *A)

AP, NPI, NPO, ADP, END

If the flowrate is obtained using a polynomial pump curve, the format is:

AP, NPI, NPO, A0, A1, A2, A3, A4, END

The following definitions apply in the above arrays:

NPI - System inlet pressure node

AW - Tabulated curve of flowrate vs time

NPO - System outlet pressure node

ADP - Tabulated pump curve giving pressure rise as a function of flowrate

A0,A1,A2,A3,A4 - Polynomial curve fit constants for flowrate as a function of pressure rise. i.e.,

$$\dot{w} = A0+A1\cdot\Delta P + A2\cdot\Delta P^2 + A3\cdot\Delta P^4 + A4\cdot\Delta P^4$$

KOP - is an integer code for checkout print from subroutine PFCS. If KOP = 1 a checkout print will be obtained. If KOP = 0 a print will not be obtained.

DYNAMIC STORAGE REQUIREMENTS:

Dynamic storage required for PFCS is $1/2(NPRN^2 + 6*NPRN+12)$, where NPRN is the maximum of the number of pressure nodes in any network.

TABLE 2 VALUE OF GC FOR VARIOUS PROBLEM UNITS

UNITS				GC
MASS	FORCE	LENGTH	TIME	
LB	LB_f	In.	Sec	386.1
↓	↓	↓	Min	$1.390X10^6$
			Hr	$5.004X10^9$
		Ft.	Sec	32.174
		↓	Min	$1.1583X10^5$
			Hr	$4.1696X10^8$
		Yd.	Sec	10.725
		↓	Min	$3.861X10^4$
			Hr	$1.3899X10^8$
GRAM	dyne	Centimeter	Sec	1.0
↓	↓	↓	Min	3600.
			Hr	$1.296X10^7$
KILOGRAM	Newton	Centimeter	Sec	1×10^{-2}
↓	↓	↓	Min	36
			Hr	$1.296X10^5$
		Meter	Sec	1.0
		↓	Min	3600.
			Hr	$1.296X10^7$

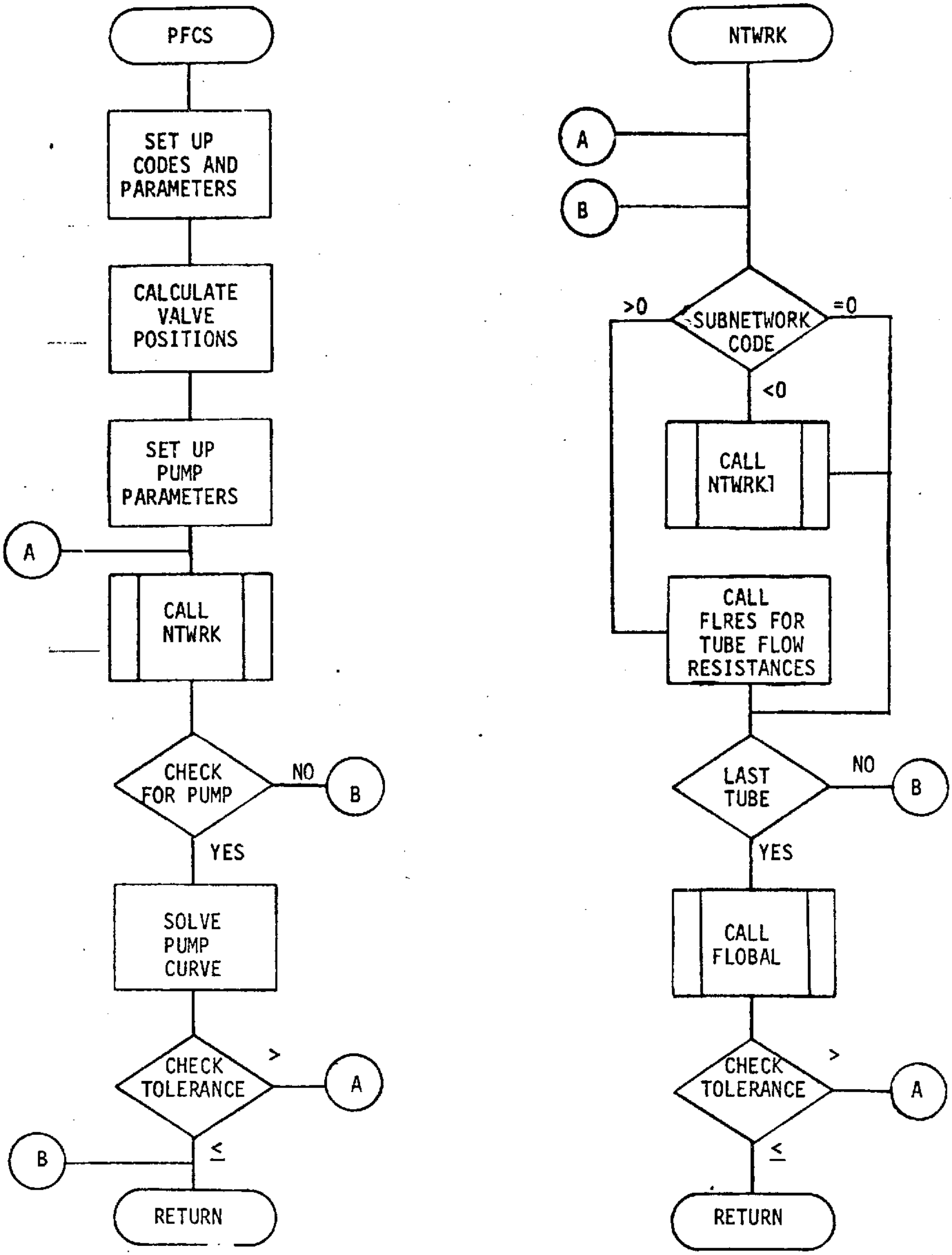


FIGURE 7 FLOW CHARTS OF PFCS AND NTWRK

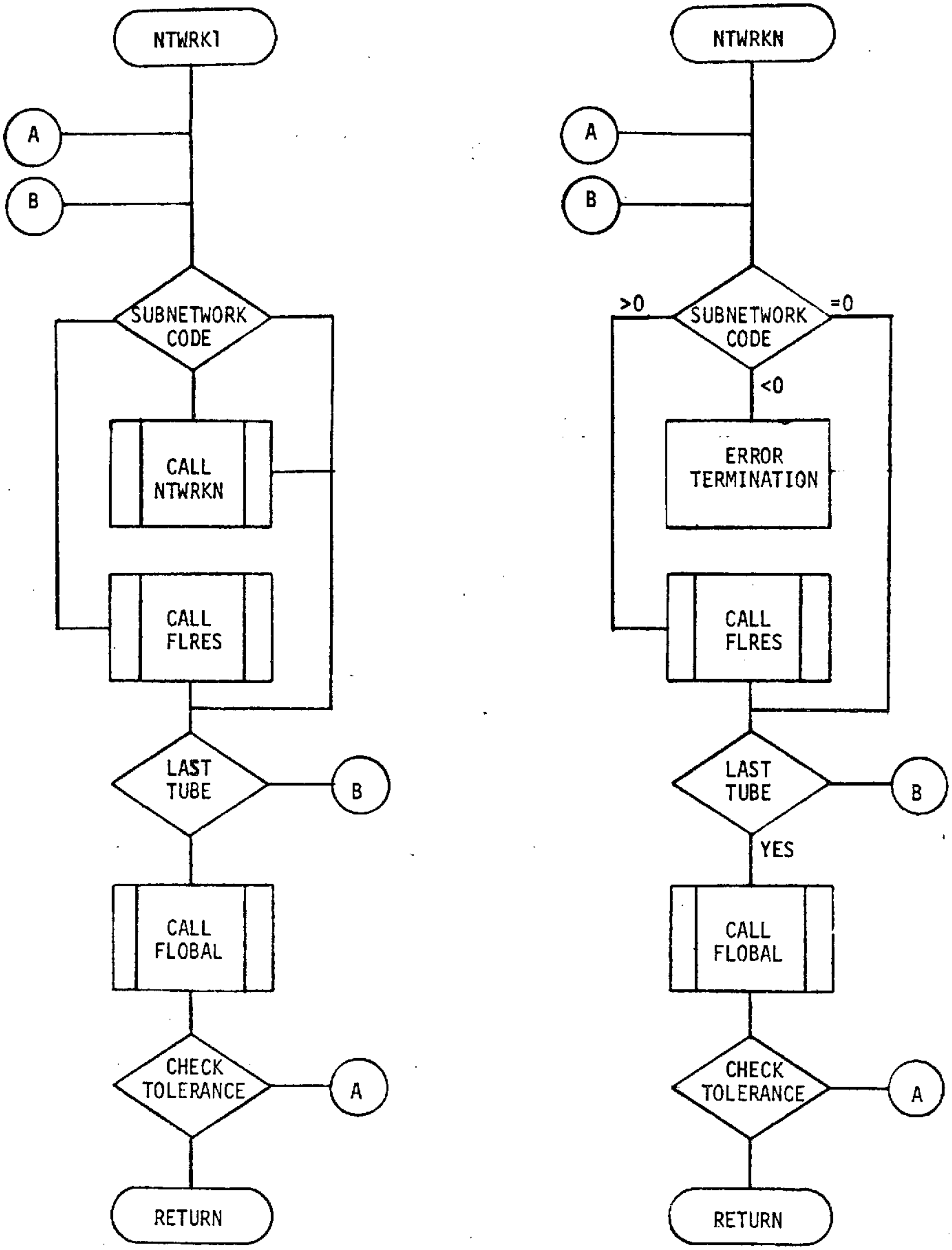


FIGURE 8 FLOW CHARTS OF NTWRK1 AND NTWRKN

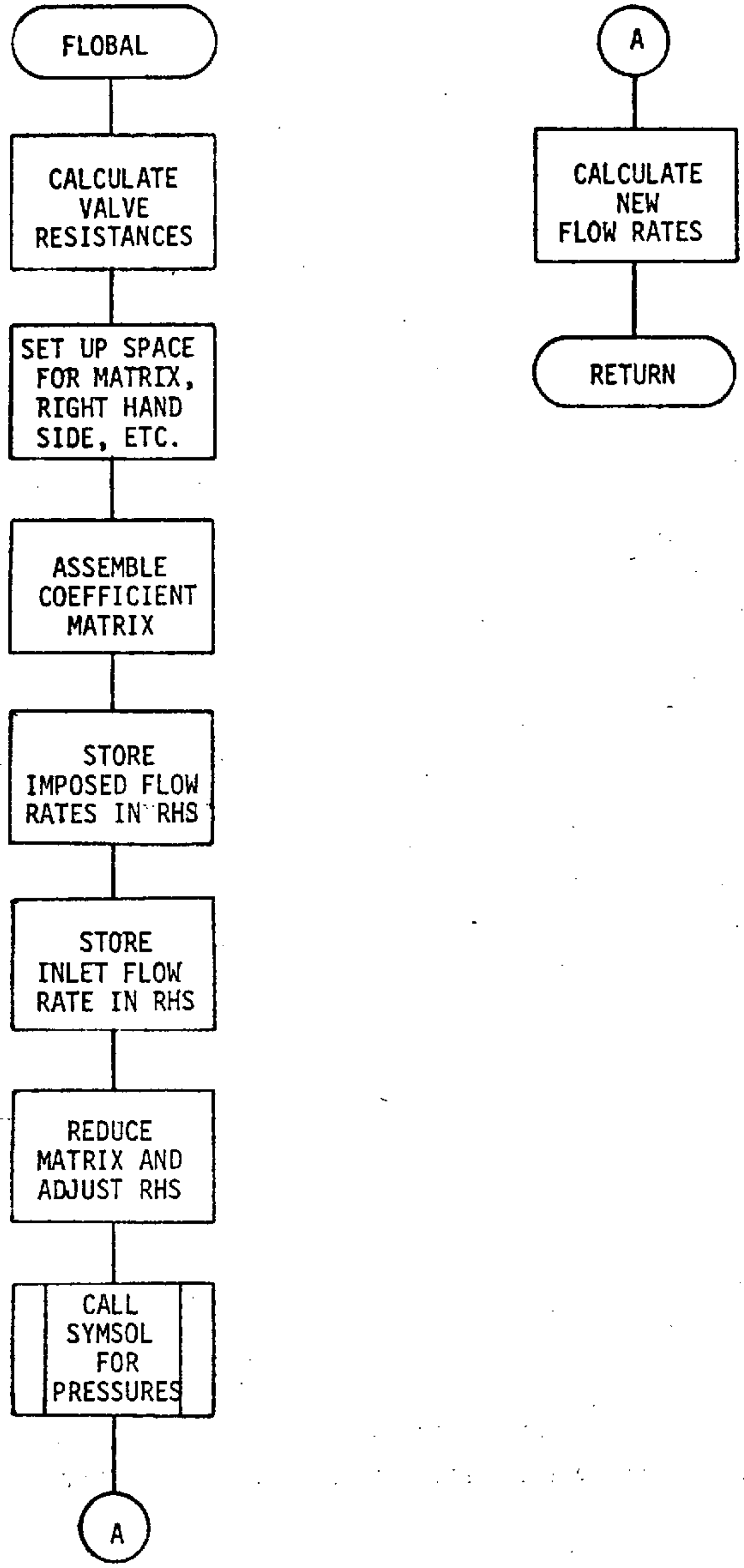


FIGURE 9 FLOW CHART OF FLOBAL

SUBROUTINE NAME: HSTRY

PURPOSE:

Subroutine HSTRY stores the problem time, the pressures of all pressure nodes, the valve positions for all valves, the flowrates for all tubes, and the temperatures of all temperature nodes at an input interval on a magnetic tape (the history tape) mounted on Unit T. The number of records written on the history tape is the number of history intervals plus two. The first record contains a title, an integer count of the number of items to be written for each of the four categories (pressures, valve positions, flowrates and temperatures), and the actual node numbers in order of the relative numbers. The second thru the next-to-last records contain the history records with one for each time point and the last record is the same as the next-to-last except the time is negative. The arguments to HSTRY are the pressure array, PR, the valve position array, VP, the flow rate array, W, and the history tape writting interval, TINC.

The format for the history tape is as follows:

Record No. 1

Title (Written Internally) in 12A6 format, 0, 0, 0, 0, 0, 0, No. of pressure nodes, number of valve positions, 0, 0, 0, number of tubes, 0, 0, number of nodes, actual node numbers in increasing order of relative node numbers.

Record No. 2

Initial problem time, pressures, valve positions, flowrates, node temperatures

Record No. 3

Second history time, pressures, valve positions, flowrates, node temperatures

⋮

Record No. N+1 (Where N = number of history time slices to be written)

Last history time, pressures, valve positions, flowrates, node temperatures

Record No. N+2

Same as last record except time is negative

RESTRICTIONS:

Should be called in VARIABLES 2. An output history tape should be mounted on Unit T. Subroutine TMCHK must be in VARIABLES 2 prior to the call to Subroutine HSTRY if TIMCHK is called in the problem.

If the backup feature is used in VARIABLES 2, the call to subroutine HSTRY should not be made until the last pass to avoid nonincreasing time records or invalid data. For example:

```
              BCD  3VARIABLES 2
                       .
                       .
                       .
         F    IF (T(16) .LT.  TMAX) BACKUP = 1.
                       .
                       .
                       .
         F    IF (BACKUP  .GT.  0.) GO TO 10
                 HSTRY (A1, A2, A3, .01)
         F 10 CONTINUE
              END
```

CALLING SEQUENCE:

HSTRY(PR(IC), VP(IC), W(IC), TINC)

PR	-	is the pressure (or pressure drop) array
VP	-	is the valve position array
W	-	is the flowrate array
TINC	-	is the time interval for plotting

SUBROUTINE NAME: NEWTMP

PURPOSE:

Subroutine NEWTMP will read the node temperatures, flowrates, pressures and valve positions at time TMPTIM from the history tape assigned to Unit U generated by subroutine HSTRY for a previous run on Unit T to initiate a problem at these conditions. The pressure array, PR, valve position array, VP, flow rate array, W, and time to read the tape, TMPTIM, are arguments. The subroutine should be called in the execution block prior to the call to the temperature solution subroutine.

RESTRICTIONS:

Must be called in the EXECUTION block prior to the call to the appropriate temperature solution subroutine. The history tape must be assigned on Unit U.

CALLING SEQUENCE:

NEWTMP(PR(IC), VP(IC), W(IC), TMPTIM)

PR	-	is the pressure array
VP	-	is the valve position array
W	-	is the flowrate array
TMPTIM	-	is the time to read the values of PR, VP, W and temperatures from the U tape

SUBROUTINE NAME: FLPRNT

PURPOSE:

Subroutine FLPRNT will write the values of the DATA array of real numbers at 10 to a line. The array is labeled by the variable input HEAD which contains 9 six character alpha numeric words. The array location of every tenth value in the array is identified to the right of the appropriate line. FLPRNT was written primarily for the output of flowrates, pressures, pressure drops, and valve positions obtained from PFCS but may be used for the output of any real array.

RESTRICTIONS:

Should be called from OUTPUT. The array must be real.

CALLING SEQUENCE:

FLPRNT(DATA(IC), HEAD(DV))

SUBROUTINE NAMES: GENOUT, GENI OR GENR

PURPOSE:

These subroutines print out arrays of numbers 10 to a line. GENOUT prints either real numbers, integer or both. GENI and GENR print integers and real number arrays respectively. The integers are written in an I9 format and the real numbers in an E12.4 format.

RESTRICTIONS:

GENI writes arrays of integers only. GENR writes arrays of real numbers only.

CALLING SEQUENCE:

```
GENOUT (A, ISTRT, ISTP, 'NAME')
GENI   (A, ISTRT, ISTP, 'NAME')
GENR   (A, ISTRT, ISTP, 'NAME')
```

where A - is the array location

ISTRT - is the first value in A being written

ISTP - is the last value in A being written

'NAME' - is a title of 22 Hollerith words for identification

SUBROUTINE NAME: FLUX

PURPOSE:

Subroutine FLUX permits doublet time variant curve values stored on magnetic tape unit NFLXTP to be read into NCRV arrays starting at array DATA when the mission time exceeds DQTIME. The flux tape must be generated prior to the run using a GE routine LTVFTP. This routine generates the flux tape in the following format:

Record No. 1

First Read Time

Record No. 2

Number of points on first curve (Integer), first curve independent variables, first curve dependent variables, number of points on second curve, second curve independent variables, second curve dependent variables, etc. for all curves.

Record No. 3

Second Read Time

Record No. 4

Same as Record No. 2 except with new values

Record No. 5

Third Read Time

Etc. until all blocks of data are on tape.

Subroutine FLUX writes the values from the appropriate NFLXTP record into the arrays defined by DATA and NCRV in the proper doublet array format. Flux values should be input into the heat flux arrays ($DATA_1$---$DATA_{NCRV}$) initially if the user doesn't want the values to be read from the tape at the start of the problem. The value of QTIME should initially be the value of the time the first read is desired.

RESTRICTIONS:

The following restrictions apply:

(1) The initial block of curve data must be input on cards or data

(2) Particular curves must have the same number of points on each block of data read in as were input on cards initially

(3) Each curve may have a different number of points

(4) The first point on each curve in each block of data must be the same as the last point on that curve in the previous block of data

(5) All incident heat curves must be in a single block by themselves.

CALLING SEQUENCE:

FLUX(NFLXTP, DATA, NCRV, DQTIME, QTIME)

where

NFLXTP - logical unit to which the flux tape is assigned. Must be supplied by a user constant.

DATA - starting location (IC) for flux curves

NCRV - number of flux curves to be updated from the flux tape

DQTIME - time scale shift for flux curves DQTIME is added to each independent value for each flux curve read from NFLXTP

QTIME - the last point on the latest set of flux curves read from NFLXTP. (QTIME = FLXTIM + DQTIME, where FLXTIM is the time read from the flux tape) must be supplied by user constant.

SUBROUTINE NAME: TIMCHK

PURPOSE:

Subroutine TIMCHK compares the elapsed computer time against the requested computer time, RTIME, and terminates the run if RTIME is exceeded by the elapsed time. If the second argument, KODE, is non-zero an output of computer time used will be printed out on each call to TIMCHK. Thus, a call to TIMCHK in VARIABLES 2 should normally be with KODE=0. If the output of computer time used is desired, TIMCHK should be called from OUTPUT with KODE ≠ 0. The most desirable procedure is to supply two calls to TIMCHK : (1) a call in VARIABLES 2 with KODE = 0 and (2) a call in OUTPUT with KODE ≠ 0.

RESTRICTIONS:

KODE should zero when called from VARIABLES 1 or 2.

CALLING SEQUENCE:

TIMCHK (RTIME, KODE)

where RTIME = maximum computer time requested

KODE = print code: = 0, computer time used is not printed out

≠ 0, computer time used is printed out on each call to TIMCHK

SUBROUTINE NAME: REVPOL

PURPOSE:

This subroutine performs single variable linear interpolation on a doublet array of X,Y pairs in the same manner as D1DEG1 except in reverse order. The array is interpolated in reverse order to obtain the value of independent variable, X, which corresponds to the input dependent variable, Y.

RESTRICTIONS:

All values must be floating point numbers.

CALLING SEQUENCE:

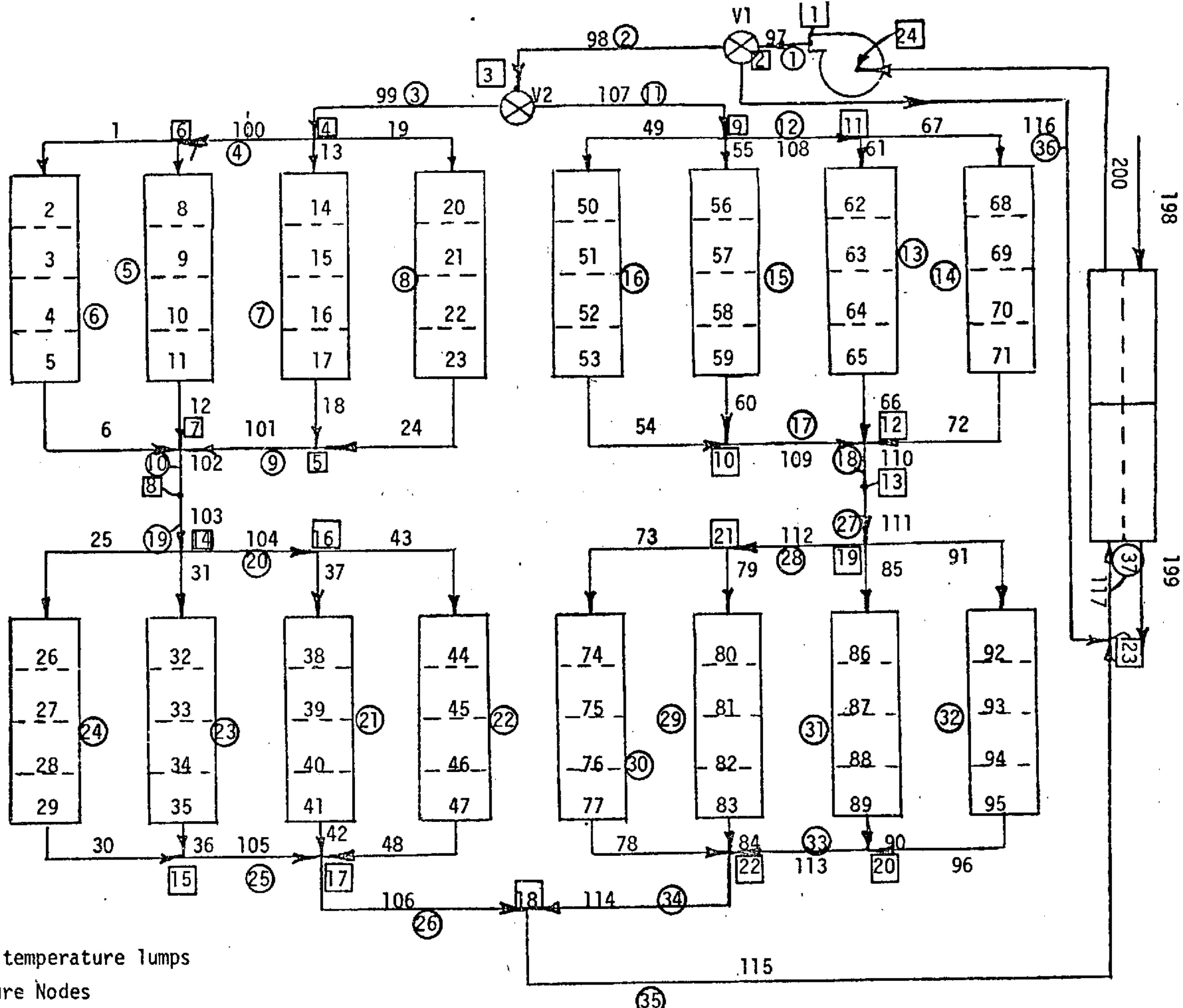
REVPOL (Y,A(IC),X)

where Y	-	input value of dependent variable
A	-	Doublet array of X,Y pairs
X	-	output value of independent variable

5.0 SAMPLE PROBLEM

A sample problem was prepared for the SINDA routine to demonstrate the input and output for a typical thermal/flow analysis problem. A schematic of the problem is shown in Figure 10 . The problem consists of 8 two dimensional radiator panels, each modeled by two flow paths (one for the main panel of 11 tubes and one for the prime bypass tube). Contained in the system are a pump, a bypass valve (valve No. 1) and a stagnation valve between the two flow paths. The heat load to the radiator system comes through a counter flow heat exchanger which has a controlled inlet temperature of 40°F. The fluid is Freon 21 in the radiator system and water on the cooled side of the heat exchanger. The nodal subdivision for the fluid system is shown in Figure 10. The structure nodal subdivision is shown in Figure 11.

The output for the run is presented in Table 3. A selected few items were plotted using the plot package described in Appendix B. These are presented in Figures 12 thru 17.



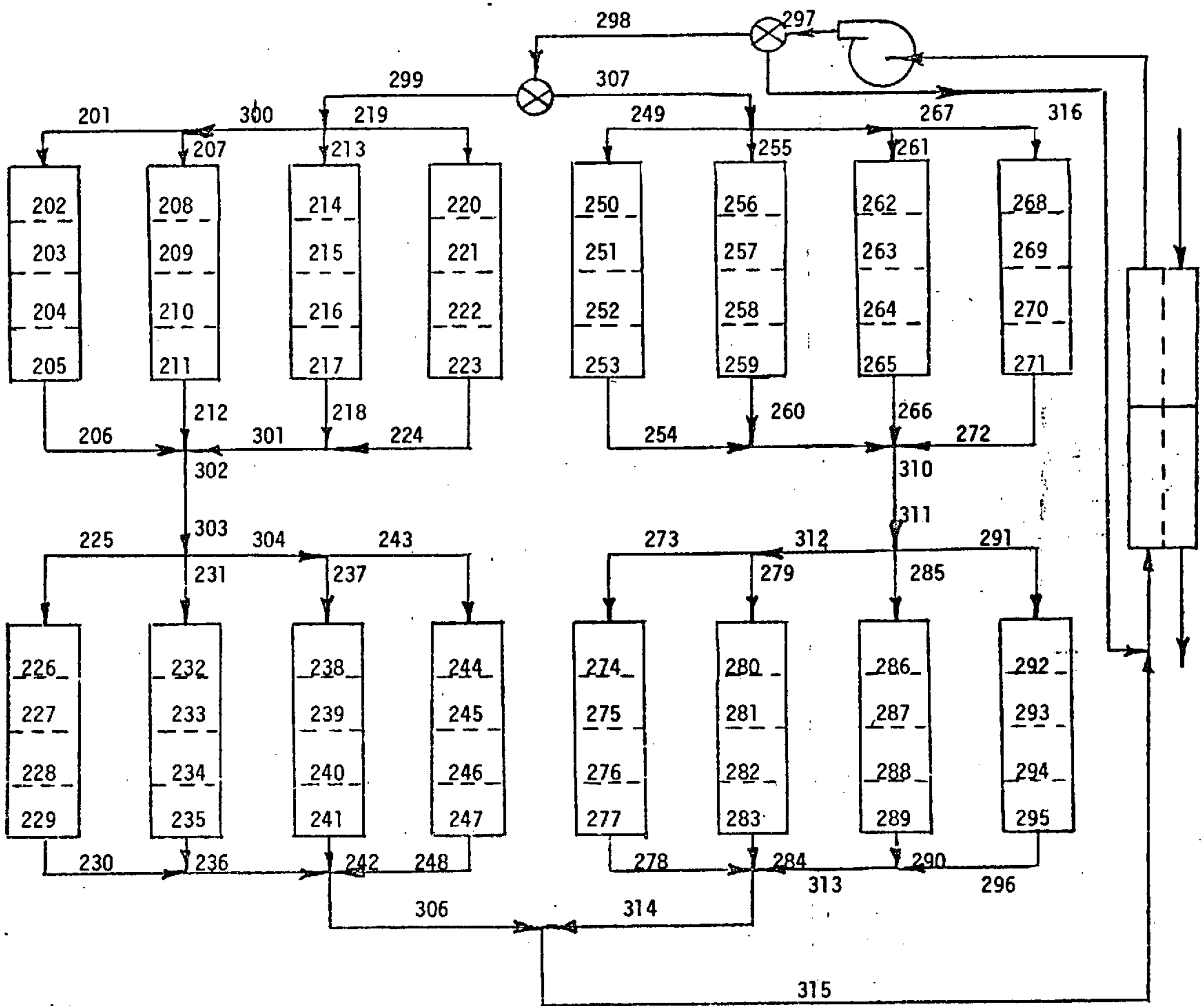
xx - Fluid temperature lumps

[x] - Pressure Nodes

(x) - Tube numbers

vx - Valve numbers

FIGURE 10 FLUID MODEL OF THE SAMPLE PROBLEM



xx - Structure Lumps

FIGURE 11 STRUCTURE MODEL FOR THE SAMPLE PROBLEM

TABLE 3

OUTPUT FOR SAMPLE PROBLEMS

```
• ASG G=A05462                                   25 FEB 73      14:14:46

• ASG Z=A05743                                   25 FEB 73      14:14:46

•W ASG T=A02293                                  25 FEB 73      14:14:46

•  ASG D,J,K,M                                   25 FEB 73      14:14:46
```

```
@ XQT CUR                                                   25 FEB 73     14:14:46
   1.  TRW 6                                                              14:14:47
   2.  IN 6                                                               14:14:47

     END OF FILE -- UNIT 6

     END CUR LCC 1102-0039 L9
@P NOG        SAMPLE PROBLEM FOR SINDA VERSION 9
```

STARTING ADDRESS 014000

CORE LIMITS 014000 047745 050366 163771 163772 163777

```
PREPRO/CODE
0   050366-050425
1   014000-014459

NOUTS /RLEC4
0   050426-050432
1   014455-015457
2   050433-050450

NIOINS/RLEC5
1   015460-015531
2   050451-050501

NTABS /CODE
0   050502-050650

NFMTS /RLEC4
1   015532-016467
2   050651-050665

NCNVTS/RLEC5
1   016470-016714
2   050666-050759

NOTINS/RLEC5
1   016715-017364
2   050755-051020

FPACKS/CODE
1   017365-017430

DEPTH /******
0   051021-051026

NFIVS /RL22
1   017431-017453

NBDFVS/RLEC4
0   051027-051213
```

```
NINPT$/RLEC5
0   051214-051216
1   017454-020571
2   051217-051254

NININ$/RLEC4
1   020572-020727
2   051255-051304

NIER$ /RLEC5
0   051305-051305
1   020730-021235
2   051306-051402

NERR$ /RLEC4
0   051403-051572
1   021236-021700

NFOUT$/RLEC5
1   021701-022132
2   051573-051574

NBUFF$/RL23
1   022133-022155
2   051575-052605

GAWND$/RLEC4
1   022156-022250

DUMSUB/CODE
0   052606-052650
1   022251-022335

NSTOP$/RLEC5
1   022336-022347

NEWS  /CODE
0   052651-052705
1   022350-022365

TITLEJ/CODE
0   052706-053006
1   022366-022445

CRDBLK/******
0   053007-054023

TAPE  /******
0   054024-054035

BLKBUF/******
0   054036-055105

TABCON/******
0   055106-055375

DATA  /******
0   055376-055417
```

```
LOGIC /******
0   055420-055519

PLOGIC/******
0   055515-055526

SADCOM/******
0   055527-056130

BUCKET/******
0   056131-156500

POINT /******
0   156501-156575

CHECKD/******
0   156576-156747

FLAGS /******
0   156750-156752

JPS   /******
0   156753-156753

CIMAGE/******
0   156754-157333

RFINP$/RLECS
1   022496-022707
2   157334-157334

SEARCH/CODE
0   157335-157351
1   022710-022774

MAUTO$/CODE
1   022775-023433

BLKCRD/CODE
0   157352-157630
1   023434-024327

WRTBLK/CODE
1   024330-024341

STFFB /CODE
0   157631-157643
1   024342-024371

FINDRM/CODE
0   157644-157717
1   024372-024723

SQUEEZ/CODE
0   157720-157742
1   024724-025015
```

```
SREADC/CODE
0   157743-160035
1   025016-025350

MREADC/CODE
0   160036-160055
1   025351-025501

RDBLK /CODE
0   160056-160070
1   025502-025610

MERRS /CODE
0   160071-160132
1   025611-025631

NTRAN /CODE
0   160133-160137
1   025632-025724

CURDRM/CODE
0   160140-160144
1   025725-026004

             s*EDIT  /CODE
                0   160145-161056
                1   026005-027374

             s*MODUR /CODE
                0   160145-160304
                1   026005-027063

              BUFTAP/CODE
                0   160305-160317
                1   027064-027223

              BUFMCF/CODE
                0   160320-160326
                1   027224-027306

              TOCENT/CODE
                0   160327-160350
                1   027307-027534

              FINDRD/CODE
                0   160351-160357
                1   027535-027613

              MPCH  /CODE
                0   160360-160372
                1   027614-027641

              AZILUN/CODE
                0   160373-160450
                1   027642-027746

              TOCOX /CODE
                0   160451-160462
```

```
  1   027747-030035

 TOCWRT/CODE
  0   160463-160571
  1   030036-030151

 RDSET /CODE
  0   160572-160601
  1   030152-030223

 MURTIT/CODE
  0   160602-160646
  1   030224-030310

 EDTTIM/CODE
  0   160647-160663
  1   030311-030430

 MSC$01/CODE
  0   160664-160705

 ETOD$ /CODE
  1   030431-030465

 CLOCK /CODE
  0   160706-160710
  1   030466-030546

 NEWMOD/CODE
  0   160711-161040
  1   030547-032045

 UPDAT2/CODE
  0   161041-161130
  1   032046-032606

 UPDAT1/CODE
  0   161131-161217
  1   032607-033613

 MWRITC/CODE
  0   161220-161275
  1   033614-034043

 COMPAS/CODE
  0   161276-161615
  1   034044-034275

 SINDAX/CODE
  0   161616-161664
  1   034276-034533

 GREAD /CODE
  0   161665-161743
  1   034534-035074

 STRTCH/CODE
  0   161744-162125
```

```
  1   035075-035613

 GETFLD/CODE
  0   162126-162245
  1   035614-035774

 MOVE  /CODE
  0   162246-162264
  1   035775-036127

 NCOL  /CODE
  0   162265-162271
  1   036130-036150

 MVCH  /CODE
  0   162272-162302
  1   036151-036212

*BUFBLK/******
  0   162303-163276

*MODNEW/******
  0   163277-163373

*MURCRD/******
  0   163374-163400

s*GENLNK/CODE
  0   160145-160267
  1   026005-026316

s*PSEUDO/CODE
  0   160145-160271
  1   026005-030131

 *PCS2  /CODE
  0   160272-160301
  1   030132-030177

s*CODERD/CODE
  0   160145-160426
  1   026005-030640

 IMBED /CODE
  0   160427-160520
  1   030641-031107

 *DATARD/CODE
  0   160521-161447
  1   031110-035727

 *ERRMES/CODE
  0   161450-162757
  1   035730-036611

 *CONVRT/CODE
  0   162760-163015
  1   036612-036737
```

```
*TYPCHK/CODE
  0   163016-163036
  1   036740-037155

*QDATA /CODE
  0   163037-163153
  1   037156-040745

*RELACT/CODE
  0   163154-163205
  1   040746-041226

*WRTDTA/CODE
  0   163206-163227
  1   041227-042064

*WRTPMT/CODE
  0   163230-163246
  1   042065-042442

 GSWAP /CODE
  0   163247-163266
  1   042443-042554

*INCORE/CODE
  0   163267-163316
  1   042555-043350

*SETFMT/CODE
  0   163317-163416
  1   043351-043537

*GENLK /CODE
  0   163417-163473
  1   043540-044147

                    **NODES /CODE
                      0   163474-163562
                      1   044150-047235

                    **CONDS /CODE
                      0   163474-163613
                      1   044150-047745

**PRESUB/CODE
   0   160145-160223
   1   026005-026133

 *SINDA*/CODE
   0   160224-161133
   1   026134-030516

 *MIT2FN/CODE
   0   161134-161226
   1   030517-031207

 *PLPINT/CODE
```

```
   0   161227-161447
   1   031210-031712

**SPLIT /CODE
   0   160145-160170
   1   026005-026770

 *SKIP  /CODE
   0   160171-160216
   1   026771-027301

END OF ALLOCATION 1103 0039A 09099
```

```
********************************************************************************************************************
******                                                                                                        ******
******         SINDA/VERSION   9 DATED FEB 12, 1973  (COMPILED UNDER UNIVAC FORTRAN/25 ON FEB 12, 1973)        ******
******                                                                                                        ******
********************************************************************************************************************

****** A DESCRIPTION OF THE MODIFICATIONS TO EACH VERSION OF SINDA IS CONTAINED ON COMMENT CARDS IN SUBROUTINE NEWS ****
```

*SINDA MODELC Z

BEGIN SINDA PREPROCESSOR, VERSION 8, AT 14:15:03

```
BCD 3THERMAL LPCS
BCD 6 SAMPLE PROBLEM FOR SINDA VERSION 9
END
BCD 3NODE DATA
REM NODE,NUM,INC , TI      ,A(T), CONST    $
SIM    1,  8,    6,  70.    ,  A3, .012096  $  FLUID LUMPS
SIM    2,  8,    6,  70.    ,  A3, .00305   $
SIM    3,  8,    6,  70.    ,  A3, .00305   $
SIM    4,  8,    6,  70.    ,  A3, .00305   $
SIM    5,  8,    6,  70.    ,  A3, .00305   $
SIM    6,  8,    6,  70.    ,  A3, .012096  $
SIM   49,  8,    6,  70.    ,  A3, .012096  $
SIM   50,  8,    6,  70.    ,  A3, .0000213 $
SIM   51,  8,    6,  70.    ,  A3, .0000213 $
SIM   52,  8,    6,  70.    ,  A3, .0000213 $
SIM   53,  8,    6,  70.    ,  A3, .0000213 $
SIM   54,  8,    6,  70.    ,  A3, .012096  $
SIV   97            ,  70.    ,  A3, .005040  $
SIV   98            ,  70.    ,  A3, .005090  $
SIV   99            ,  70.    ,  A3, .020160  $
SIV  100            ,  70.    ,  A3, .007056  $
SIV  101            ,  70.    ,  A3, .007056  $
SIV  102            ,  70.    ,  A3, .002520  $
SIV  103            ,  70.    ,  A3, .002520  $
SIV  104            ,  70.    ,  A3, .007056  $
SIV  105            ,  70.    ,  A3, .007056  $
SIV  106            ,  70.    ,  A3, .020160  $
SIV  107            ,  70.    ,  A3, .020160  $
SIV  108            ,  70.    ,  A3, .005040  $
SIV  109            ,  70.    ,  A3, .005040  $
SIV  110            ,  70.    ,  A3, .002520  $
SIV  111            ,  70.    ,  A3, .002520  $
SIV  112            ,  70.    ,  A3, .005090  $
SIV  113            ,  70.    ,  A3, .005040  $
SIV  114            ,  70.    ,  A3, .020160  $
SIV  115            ,  70.    ,  A3, .005090  $
SIV  116            ,  70.    ,  A3, .002016  $
SIV  117            ,  70.    ,  A3, .050400  $
    -198           ,100.    ,  1.0
    -199           , 40.    ,  1.0
    -200           , 100.   ,  1.0           $
SIM  201,   8,    6,    70.   ,  A4,  .720    $      TUBE LUMPS
SIM  202,   8,    6,    70.   ,  A4,10.3      $
SIM  203,   8,    6,    70.   ,  A4,10.3      $
SIM  204,   8,    6,    70.   ,  A4,10.3      $
SIM  205,   8,    6,    70.   ,  A4,10.3      $
SIM  206,   8,    6,    70.   ,  A4,  .720    $
SIM  249,   8,    6,    70.   ,  A4,  .720    $
SIM  250,   8,    6,    70.   ,  A4,  .0718   $
SIM  251,   8,    6,    70.   ,  A4,  .0718   $
SIM  252,   8,    6,    70.   ,  A4,  .0718   $
SIM  253,   8,    6,    70.   ,  A4,  .0718   $
SIM  254,   8,    6,    70.   ,  A4,  .720    $
SIV  297             ,  70.   ,  A4,  .299    $
SIV  298             ,  70.   ,  A4,  .299    $
```

```
SIV  299      ,  70.   ,  A4, 1.20     $
SIV  300      ,  70.   ,  A4,  .419    $
SIV  301      ,  70.   ,  A4,  .419    $
SIV  302      ,  70.   ,  A4,  .150    $
SIV  303      ,  70.   ,  A4,  .150    $
SIV  304      ,  70.   ,  A4,  .419    $
SIV  305      ,  70.   ,  A4,  .419    $
SIV  306      ,  70.   ,  A4,  .720    $
SIV  307      ,  70.   ,  A4, 1.200    $
SIV  308      ,  70.   ,  A4,  .720    $
SIV  309      ,  70.   ,  A4,  .720    $
SIV  310      ,  70.   ,  A4,  .150    $
SIV  311      ,  70.   ,  A4,  .150    $
SIV  312      ,  70.   ,  A4,  .720    $
SIV  313      ,  70.   ,  A4,  .720    $
SIV  314      ,  70.   ,  A4, 1.200    $
SIV  315      ,  70.   ,  A4,  .299    $
SIV  316      ,  70.   ,  A4,  .012    $
SIV  317      ,  70.   ,  A4, 2.99     $
    -400      , -459.69 ,1.0           $
END
```

RELATIVE NODE NUMBERS	ACTUAL NODE NUMBERS									
1 THRU 10	1	7	13	19	25	31	37	43	2	8
11 THRU 20	14	20	26	32	38	44	3	9	15	21
21 THRU 30	27	33	39	45	4	10	16	22	28	34
31 THRU 40	40	46	5	11	17	23	29	35	41	47
41 THRU 50	6	12	18	24	30	36	42	48	49	55
51 THRU 60	61	67	73	79	85	91	50	56	62	68
61 THRU 70	74	80	86	92	51	57	63	69	75	81
71 THRU 80	87	93	52	58	64	70	76	82	88	94
81 THRU 90	53	59	65	71	77	83	89	95	54	60
91 THRU 100	66	72	78	84	90	96	97	98	99	100
101 THRU 110	101	102	103	104	105	106	107	108	109	110
111 THRU 120	111	112	113	114	115	116	117	201	207	213
121 THRU 130	219	225	231	237	243	202	208	214	220	226
131 THRU 140	232	238	244	203	209	215	221	227	233	239
141 THRU 150	245	204	210	216	222	228	234	240	246	205
151 THRU 160	211	217	223	229	235	241	247	206	212	218
161 THRU 170	224	230	236	242	248	249	255	261	267	273
171 THRU 180	279	285	291	250	256	262	268	274	280	286
181 THRU 190	292	251	257	263	269	275	281	287	293	252
191 THRU 200	258	264	270	276	282	288	294	253	259	265
201 THRU 210	271	277	283	289	295	254	260	266	272	278
211 THRU 220	284	290	296	297	298	299	300	301	302	303
221 THRU 230	304	305	306	307	308	309	310	311	312	313
231 THRU 238	314	315	316	317	198	199	200	400		

NODE ANALYSIS... DIFFUSION = 234, ARITHMETIC = 0, BOUNDARY = 4, TOTAL = 238

```
BCD 3SOURCE DATA
REM NODE,A(TIME),CONST $
SIT  202, A15,  16.270 $
SIT  203, A15,  16.270 $
SIT  204, A15,  16.270 $
SIT  205, A15,  16.270 $
SIT  208, A15,  16.270 $
SIT  209, A15,  16.270 $
```

```
SIT  210, A15,  16.270 $
SIT  211, A15,  16.270 $
SIT  214, A15,  16.270 $
SIT  215, A15,  16.270 $
SIT  216, A15,  16.270 $
SIT  217, A15,  16.270 $
SIT  220, A15,  16.270 $
SIT  221, A15,  16.270 $
SIT  222, A15,  16.270 $
SIT  223, A15,  16.270 $
SIT  226, A15,  16.270 $
SIT  227, A15,  16.270 $
SIT  228, A15,  16.270 $
SIT  229, A15,  16.270 $
SIT  232, A15,  16.270 $
SIT  233, A15,  16.270 $
SIT  234, A15,  16.270 $
SIT  235, A15,  16.270 $
SIT  238, A15,  16.270 $
SIT  239, A15,  16.270 $
SIT  240, A15,  16.270 $
SIT  241, A15,  16.270 $
SIT  244, A15,  16.270 $
SIT  245, A15,  16.270 $
SIT  246, A15,  16.270 $
SIT  247, A15,  16.270 $
SIT  250, A15,   0.431 $
SIT  251, A15,   0.431 $
SIT  252, A15,   0.431 $
SIT  253, A15,   0.431 $
SIT  256, A15,   0.431 $
SIT  257, A15,   0.431 $
SIT  258, A15,   0.431 $
SIT  259, A15,   0.431 $
SIT  262, A15,   0.431 $
SIT  263, A15,   0.431 $
SIT  264, A15,   0.431 $
SIT  265, A15,   0.431 $
SIT  268, A15,   0.431 $
SIT  269, A15,   0.431 $
SIT  270, A15,   0.431 $
SIT  271, A15,   0.431 $
SIT  274, A15,   0.431 $
SIT  275, A15,   0.431 $
SIT  276, A15,   0.431 $
SIT  277, A15,   0.431 $
SIT  280, A15,   0.431 $
SIT  281, A15,   0.431 $
SIT  282, A15,   0.431 $
SIT  283, A15,   0.431 $
SIT  286, A15,   0.431 $
SIT  287, A15,   0.431 $
SIT  288, A15,   0.431 $
SIT  289, A15,   0.431 $
SIT  292, A15,   0.431 $
SIT  293, A15,   0.431 $
SIT  294, A15,   0.431 $
SIT  295, A15,   0.431 $
```

```
END
BCD 3CONDUCTOR DATA
REM    NG  NOG   IG   NA  INA   NB  INB    G
GEN    1,   5,   1,  -1,   1,   2,   1,   75.         $FLOW
GEN    6,   5,   1,  -7,   1,   8,   1,   75.         $
GEN   11,   5,   1, -13,   1,  14,   1,   75.         $
GEN   16,   5,   1, -19,   1,  20,   1,   75.         $
GEN   21,   5,   1, -25,   1,  26,   1,   75.         $
GEN   26,   5,   1, -31,   1,  32,   1,   75.         $
GEN   31,   5,   1, -37,   1, 38,    1,   75.         $
GEN   36,   5,   1, -43,   1,  44,   1,   75.         $
GEN   41,   5,   1, -49,   1,  50,   1,   75.         $
GEN   46,   5,   1, -55,   1,  56,   1,   75.         $
GEN   51,   5,   1, -61,   1,  62,   1,   75.         $
GEN   56,   5,   1, -67,   1,  68,   1,   75.         $
GEN   61,   5,   1, -73,   1,  74,   1,   75.         $
GEN   66,   5,   1, -79,   1,  80,   1,   75.         $
GEN   71,   5,   1, -85,   1,  86,   1,   75.         $
GEN   76,   5,   1, -91,   1,  92,   1,   75.         $
      81, 200,  97, 625.  $
      82, -97,  98, 600.  $
      83, -97, 116,  25.  $
      84, -98,  99, 300.  $
      85, -98, 107, 300.  $
      86, -99,  13,  75.  $
      87, -99,  19,  75.  $
      88, -99, 100, 150.  $
      89,-100,   1,  75.  $
      90,-100,   7,  75.  $
      91, -18, 101,  75.  $
      92, -24, 101,  75.  $
      93,  -6, 102,  75.  $
      94, -12, 102,  75.  $
      95,-101, 102, 150.  $
      96,-102, 103, 300.  $
      97,-103,  25,  75.  $
      98,-103,  31,  75.  $
      99,-103, 104, 150.  $
     100,-104,  37,  75.  $
     101,-104,  43,  75.  $
     102, -42, 106,  75.  $
     103, -48, 106,  75.  $
     104,-105, 106, 150.  $
     105,-107,  49, 300.  $
     106,-107,  55,  75.  $
     107,-107, 108, 150.  $
     108,-108,  61,  75.  $
     109,-108,  67,  75.  $
     110, -54, 109,  75.  $
     111, -60, 109,  75.  $
     112, -66, 110,  75.  $
     113, -72, 110,  75.  $
     114,-109, 110, 150.  $
     115,-110, 111, 300.  $
     116,-111,  85,  75.  $
     117,-111,  91,  75.  $
     118,-111, 112, 150.  $
     119,-112,  73,  75.  $
```

Flow conductors identified in array 17, page 114*

*Comments added to listing of input data to clarify input of flow systems

```
        120,-112,  79,  75.  $
        121, -90, 113,  75.  $
        122, -96, 113,  75.  $
        123, -78, 114,  75.  $
        124, -84, 114,  75.  $
        125,-113, 114, 150.  $
        126,-106, 115, 300.  $
        127,-114, 115, 300.  $
        128,-115, 117, 300.  $
        129,-116, 117,  25.  $
        130, -30, 105,  75.  $
        131, -36, 105,  75.  $
GEN   201, 117,   1,   1,   1, 201,   1,      100.      $ CONVECTION
REM   NG   NOG   IG    NA  INA   NB  INB   G
GEN -401,     8,   1, 202,    6, 400,    0, 2.59E-8     $ RADIATION
GEN -409,     8,   1, 203,    6, 400,    0, 2.59E-8     $
GEN -417,     8,   1, 204,    6, 400,    0, 2.59E-8     $
GEN -425,     8,   1, 205,    6, 400,    0, 2.59E-8     $
GEN -433,     8,   1, 250,    6, 400,    0, 0.68E-9     $
GEN -441,     8,   1, 251,    6, 400,    0, 0.68E-9     $
GEN -449,     8,   1, 252,    6, 400,    0, 0.68E-9     $
GEN -457,     8,   1, 253,    6, 400,    0, 0.68E-9     $
END
```

- Connection conductor identified in array 16, page 112

RELATIVE CONDUCTOR NUMBERS			ACTUAL CONDUCTOR NUMBERS									
1	THRU	10	1	2	3	4	5	6	7	8	9	10
11	THRU	20	11	12	13	14	15	16	17	18	19	20
21	THRU	30	21	22	23	24	25	26	27	28	29	30
31	THRU	40	31	32	33	34	35	36	37	38	39	40
41	THRU	50	41	42	43	44	45	46	47	48	49	50
51	THRU	60	51	52	53	54	55	56	57	58	59	60
61	THRU	70	61	62	63	64	65	66	67	68	69	70
71	THRU	80	71	72	73	74	75	76	77	78	79	80
81	THRU	90	81	82	83	84	85	86	87	88	89	90
91	THRU	100	91	92	93	94	95	96	97	98	99	100
101	THRU	110	101	102	103	104	105	106	107	108	109	110
111	THRU	120	111	112	113	114	115	116	117	118	119	120
121	THRU	130	121	122	123	124	125	126	127	128	129	130
131	THRU	140	131	201	202	203	204	205	206	207	208	209
141	THRU	150	210	211	212	213	214	215	216	217	218	219
151	THRU	160	220	221	222	223	224	225	226	227	228	229
161	THRU	170	230	231	232	233	234	235	236	237	238	239
171	THRU	180	240	241	242	243	244	245	246	247	248	249
181	THRU	190	250	251	252	253	254	255	256	257	258	259
191	THRU	200	260	261	262	263	264	265	266	267	268	269
201	THRU	210	270	271	272	273	274	275	276	277	278	279
211	THRU	220	280	281	282	283	284	285	286	287	288	289
221	THRU	230	290	291	292	293	294	295	296	297	298	299
231	THRU	240	300	301	302	303	304	305	306	307	308	309
241	THRU	250	310	311	312	313	314	315	316	317	401	402
251	THRU	260	403	404	405	406	407	408	409	410	411	412
261	THRU	270	413	414	415	416	417	418	419	420	421	422
271	THRU	280	423	424	425	426	427	428	429	430	431	432
281	THRU	290	433	434	435	436	437	438	439	440	441	442
291	THRU	300	443	444	445	446	447	448	449	450	451	452
301	THRU	310	453	454	455	456	457	458	459	460	461	462
311	THRU	312	463	464								

```
              SAMPLE PROBLEM FOR SINDA VERSION 9                              DATE  250273  PAGE      16
CONDUCTOR ANALYSIS...  LINEAR =   248,  RADIATION =    64,  TOTAL =   312,  CONNECTIONS =   312

BCD 3CONSTANTS DATA    $
    TIMEND,3.0         $
    DTIMEI,0.01        $
    NLOOP ,100         $
    DRLXCA,0.01        $
    ARLXCA,0.01        $
    OUTPUT,1.0
    1,4.0
END                    $
CONSTANTS ANALYSIS...  USER =     1,  ADDED =    66        0     196,  TOTAL =   131

BCD 3ARRAY DATA
     1     $ FREON-21 SPECIFIC HEAT
    -400.  , .223    , -218.   , .223    , -217.   ,3.723
    -212.  ,3.723    , -211.   , .223    , -160.   , .224
    -110.  , .228    ,  -60.   , .231    ,    0.   , .237
      40.  , .244    ,   90.   , .254    ,  120.   , .264
     140.  , .274    ,  150.   , .280    ,  180.   , .295
     246.  , .315
    END
     2     $ FREON-21 DENSITY
    -400.  , 110.    , -218.   , 110.    , -217.   , 110.
    -212.  , 110.    , -211.   , 110.    , -160.   , 104.
    -110.  ,  99.25  ,  -60.   ,  96.    ,    0.   ,  91.5
      40.  ,  88.5   ,   90.   ,  84.2   ,  120.   ,  81.8
     140.  ,  80.1   ,  150.   ,  79.9   ,  180.   ,  76.
     246.  ,  69.
    END
     3     $ FREON-21 DENSITY TIMES SPECIFIC HEAT
    -400.  ,  24.53  , -218.   ,  24.53  , -217.   , 409.53
    -212.  , 409.53  , -211.   ,  24.53  , -160.   ,  23.30
    -110.  ,  22.63  ,  -60.   ,  22.18  ,    0.   ,  21.69
      40.  ,  21.59  ,   90.   ,  21.39  ,  120.   ,  21.60
     140.  ,  21.95  ,  150.   ,  22.37  ,  180.   ,  22.42
     246.  ,  21.73
    END
     4     $ ALUMINUM SPECIFIC HEAT
    -400.  ,  .092   , -300.   , .124    , -200.   , .152
    -100.  ,  .175   ,    0.   , .192    ,  100.   , .204
     200.  ,  .214
    END
     5     $ FREON-21 VISCOSITY
    -400.  ,  19.1   , -212.   ,  19.1   , -211.   ,  19.1
    -209.  ,  18.5   , -206.   ,  16.55  , -203.   ,  14.75
    -200.  ,  13.7   , -194.   ,  11.5   , -191.   ,  10.8
    -188.  ,  10.08  , -184.   ,   9.25  , -178.   ,  18.1
    -172.  ,   7.12  , -166.   ,   6.36  , -160.   ,   5.72
    -154.  ,   5.21  , -148.   ,   4.75  , -142.   ,   4.32
    -136.  ,   3.96  , -130.   ,  3.68   , -124.   ,  3.42
    -118.  ,   3.16  , -112.   ,   2.81  ,  -76.   ,   2.02
     -49.  ,   1.62  ,    0.   ,   1.17  ,   30.   ,    .994
      60.  ,    .870 ,  100.   ,    .726 ,  160.   ,    .561
     260.  ,    .396
    END
     6     $ FREON-21 THERMAL CONDUCTIVITY
    -400.  , 0.14    , 0.0     , 0.075   , 250.    , 0.035
```

```
        END
          7      $ EMISSIVITY
         -400.  , .92      , 260.     , 0.92     ,END
          11     $ INLET TEMPERATURE VS TIME
            0.  ,  80.   ,  20.     ,  80.     ,END
          12     $ INLET FLOW RATE VS TIME
            0.  , 2500.   ,  20.     , 2500.    ,END
          13     $ PUMP CURVE
           1000.0, 175000.0
           2000.0, 155660.0
           3000.0, 100000.0
           4000.0,  25000.0, END
          15     $ PANEL HEAT FLUX VS TIME
            0.  ,  40.    ,  70.     ,  40.     ,END
          16     $ CONVECTION  DATA ,ADAT
     REM  GNO , AHT ,TUBE,FLMP,TYPE, X     , F1 , F2  $ (FLMP)=FLUID LUMP
         *G201, 1.35  , 6,*T1  , 1, 12.   , 1.0, 1.0 $ G,AHT,TUB,FL,TYP
         *G202, 1.17  , 6,*T2  , 2, 15.25 , 1.0, 1.0 $    X,F1,F2
         *G203, 1.17  , 6,*T3  , 2, 18.5  , 1.0, 1.0 $
         *G204, 1.17  , 6,*T4  , 2, 21.75 , 1.0, 1.0 $
         *G205, 1.17  , 6,*T5  , 2, 24.0  , 1.0, 1.0 $
         *G206, 1.35  , 6,*T6  , 1, 36.0  , 1.0, 1.0 $
         *G207,  .5625, 5,*T7  , 3,  5.   , 1.0, 1.0 $
         *G208, 1.17  , 5,*T8  , 2,  8.25 , 1.0, 1.0 $
         *G209, 1.17  , 5,*T9  , 2, 11.5  , 1.0, 1.0 $
         *G210, 1.17  , 5,*T10 , 2, 14.75 , 1.0, 1.0 $
         *G211, 1.17  , 5,*T11 , 2, 18.0  , 1.0, 1.0 $
         *G212,  .5625, 5,*T12 , 3, 23.0  , 1.0, 1.0 $
         *G213,  .5625, 7,*T13 , 3,  5.   , 1.0, 1.0 $
         *G214, 1.17  , 7,*T14 , 2,  8.25 , 1.0, 1.0 $
         *G215, 1.17  , 7,*T15 , 2, 11.5  , 1.0, 1.0 $
         *G216, 1.17  , 7,*T16 , 2, 14.75 , 1.0, 1.0 $
         *G217, 1.17  , 7,*T17 , 2, 18.0  , 1.0, 1.0 $
         *G218,  .5625, 7,*T18 , 3, 23.0  , 1.0, 1.0 $
         *G219, 1.35  , 8,*T19 , 1, 12.0  , 1.0, 1.0 $
         *G220, 1.17  , 8,*T20 , 2, 15.25 , 1.0, 1.0 $
         *G221, 1.17  , 8,*T21 , 2, 18.5  , 1.0, 1.0 $
         *G222, 1.17  , 8,*T22 , 2, 21.75 , 1.0, 1.0 $
         *G223, 1.17  , 8,*T23 , 2, 24.0  , 1.0, 1.0 $
         *G224, 1.35  , 8,*T24 , 1, 36.0  , 1.0, 1.0 $
         *G225, 1.35  ,24,*T25 , 1, 12.0  , 1.0, 1.0 $
         *G226, 1.17  ,24,*T26 , 2, 15.25 , 1.0, 1.0 $
         *G227, 1.17  ,24,*T27 , 2, 18.5  , 1.0, 1.0 $
         *G228, 1.17  ,24,*T28 , 2, 21.75 , 1.0, 1.0 $
         *G229, 1.17  ,24,*T29 , 2, 24.0  , 1.0, 1.0 $
         *G230, 1.35  ,24,*T30 , 1, 36.0  , 1.0, 1.0 $
         *G231,  .5625,23,*T31 , 3,  5.0  , 1.0, 1.0 $
         *G232, 1.17  ,23,*T32 , 2,  8.25 , 1.0, 1.0 $
         *G233, 1.17  ,23,*T33 , 2, 11.5  , 1.0, 1.0 $
         *G234, 1.17  ,23,*T34 , 2, 14.75 , 1.0, 1.0 $
         *G235, 1.17  ,23,*T35 , 2, 18.0  , 1.0, 1.0 $
         *G236,  .5625,23,*T36 , 3, 23.0  , 1.0, 1.0 $
         *G237,  .5625,21,*T37 , 3,  5.0  , 1.0, 1.0 $
         *G238, 1.17  ,21,*T38 , 2,  8.25 , 1.0, 1.0 $
         *G239, 1.17  ,21,*T39 , 2, 11.5  , 1.0, 1.0 $
         *G240, 1.17  ,21,*T40 , 2, 14.75 , 1.0, 1.0 $
         *G241, 1.17  ,21,*T41 , 2, 18.0  , 1.0, 1.0 $
         *G242,  .5625,21,*T42 , 3, 23.0  , 1.0, 1.0 $
```

ADAT array for convection conductors;described on page 49 an the last argument to subroutine CONV1 on page 121

```
*G243, 1.35  ,22,*T43 , 1, 12.0  , 1.0, 1.0 $
*G244, 1.17  ,22,*T44 , 2, 15.25 , 1.0, 1.0 $
*G245, 1.17  ,22,*T45 , 2, 18.5  , 1.0, 1.0 $
*G246, 1.17  ,22,*T46 , 2, 21.75 , 1.0, 1.0 $
*G247, 1.17  ,22,*T47 , 2, 24.0  , 1.0, 1.0 $
*G248, 1.35  ,22,*T48 , 1, 36.0  , 1.0, 1.0 $
*G249, 1.35  ,16,*T49 , 1, 12.0  , 1.0, 1.0 $
*G250,  .0082,16,*T50 , 4, 12.25 , 1.0, 1.0 $
*G251,  .0082,16,*T51 , 4, 12.5  , 1.0, 1.0 $
*G252,  .0082,16,*T52 , 4, 12.75 , 1.0, 1.0 $
*G253,  .0082,16,*T53 , 4, 13.0  , 1.0, 1.0 $
*G254, 1.35  ,16,*T54 , 1, 25.0  , 1.0, 1.0 $
*G255,  .5625,15,*T55 , 3,  5.0  , 1.0, 1.0 $
*G256,  .0082,15,*T56 , 4,  5.25 , 1.0, 1.0 $
*G257,  .0082,15,*T57 , 4,  5.50 , 1.0, 1.0 $
*G258,  .0082,15,*T58 , 4,  5.75 , 1.0, 1.0 $
*G259,  .0082,15,*T59 , 4,  6.0  , 1.0, 1.0 $
*G260,  .5625,15,*T60 , 3, 11.0  , 1.0, 1.0 $
*G261,  .5625,13,*T61 , 3,  5.0  , 1.0, 1.0 $
*G262,  .0082,13,*T62 , 4,  5.25 , 1.0, 1.0 $
*G263,  .0082,13,*T63 , 4,  5.50 , 1.0, 1.0 $
*G264,  .0082,13,*T64 , 4,  5.75 , 1.0, 1.0 $
*G265,  .0082,13,*T65 , 4,  6.0  , 1.0, 1.0 $
*G266,  .5625,13,*T66 , 3, 11.0  , 1.0, 1.0 $
*G267, 1.35  ,14,*T67 , 1, 12.0  , 1.0, 1.0 $
*G268,  .0082,14,*T68 , 4, 12.25 , 1.0, 1.0 $
*G269,  .0082,14,*T69 , 4, 12.50 , 1.0, 1.0 $
*G270,  .0082,14,*T70 , 4, 12.75 , 1.0, 1.0 $
*G271,  .0082,14,*T71 , 4, 13.0  , 1.0, 1.0 $
*G272, 1.35  ,13,*T72 , 1, 25.0  , 1.0, 1.0 $
*G273, 1.35  ,30,*T73 , 1, 12.0  , 1.0, 1.0 $
*G274,  .0082,30,*T74 , 4, 12.25 , 1.0, 1.0 $
*G275,  .0082,30,*T75 , 4, 12.5  , 1.0, 1.0 $
*G276,  .0082,30,*T76 , 4, 12.75 , 1.0, 1.0 $
*G278, 1.35  ,30,*T78 , 1, 25.0  , 1.0, 1.0 $
*G279,  .5625,29,*T79 , 3,  5.   , 1.0, 1.0 $
*G280,  .0082,29,*T80 , 4,  5.25 , 1.0, 1.0 $
*G281,  .0082,29,*T81 , 4,  5.5  , 1.0, 1.0 $
*G282,  .0082,29,*T82 , 4,  5.75 , 1.0, 1.0 $
*G283,  .0082,29,*T83 , 4,  6.0  , 1.0, 1.0 $
*G284,  .5625,29,*T84 , 3, 11.0  , 1.0, 1.0 $
*G285,  .5625,31,*T85 , 3,  5.   , 1.0, 1.0 $
*G286,  .0082,31,*T86 , 4,  5.25 , 1.0, 1.0 $
*G287,  .0082,31,*T87 , 4,  5.5  , 1.0, 1.0 $
*G288,  .0082,31,*T88 , 4,  5.75 , 1.0, 1.0 $
*G289,  .0082,31,*T89 , 4,  6.0  , 1.0, 1.0 $
*G290,  .5625,31,*T90 , 3, 11.0  , 1.0, 1.0 $
*G291, 1.35  ,32,*T91 , 1, 12.0  , 1.0, 1.0 $
*G292,  .0082,32,*T92 , 4, 12.25 , 1.0, 1.0 $
*G293,  .0082,32,*T93 , 4, 12.5  , 1.0, 1.0 $
*G294,  .0082,32,*T94 , 4, 12.75 , 1.0, 1.0 $
*G295,  .0082,32,*T95 , 4, 13.0  , 1.0, 1.0 $
*G296, 1.35  ,32,*T96 , 1, 25.0  , 1.0, 1.0 $
*G297,  .5625, 1,*T97 , 3,  5.0  , 1.0, 1.0 $
*G298,  .5625, 2,*T98 , 3,  5.0  , 1.0, 1.0 $
*G299, 2.25  , 3,*T99 , 5, 20.0  , 1.0, 1.0 $
*G300,  .7875, 4,*T100, 8,  7.0  , 1.0, 1.0 $
*G301,  .7875, 9,*T101, 8,  7.0  , 1.0, 1.0 $
```

ADAT array (Cont'd)

```
          SAMPLF PROBLEM FOR SINDA VERSION 9
*G302,  .281 ,10,*T102, 6,  2.5  , 1.0, 1.0 $
*G303,  .281 ,19,*T103, 6,  2.5  , 1.0, 1.0 $
*G304,  .7875,20,*T104, 8,  7.0  , 1.0, 1.0 $
*G305,  .7875,25,*T105, 8,  7.0  , 1.0, 1.0 $
*G306, 2.25  ,26,*T106, 5, 20.0  , 1.0, 1.0 $
*G307, 2.25  ,11,*T107, 5, 20.0  , 1.0, 1.0 $
*G308,  .7875,12,*T108, 8,  7.0  , 1.0, 1.0 $
*G309,  .7875,17,*T109, 8,  7.0  , 1.0, 1.0 $
*G310,  .281 ,18,*T110, 6,  2.5  , 1.0, 1.0 $
*G311,  .281 ,27,*T111, 6,  2.5  , 1.0, 1.0 $
*G312,  .7875,28,*T112, 8,  7.0  , 1.0, 1.0 $
*G313,  .7875,33,*T113, 8,  7.0  , 1.0, 1.0 $
*G314, 2.25  ,34,*T114, 5, 20.0  , 1.0, 1.0 $
*G315, 5.62  ,35,*T115, 7, 50.0  , 1.0, 1.0 $
*G316,  .225 ,36,*T116, 9,  2.0  , 1.0, 1.0 $
*G317,  .5625,37,*T117, 3,  5.0  , 1.0, 1.0 $
END
 17     $ FLOW CONDUCTOR DATA
*G1  ,*T1  ,  6 $
*G2  ,*T2  ,  6 $
*G3  ,*T3  ,  6 $
*G4  ,*T4  ,  6 $
*G5  ,*T5  ,  6 $
*G6  ,*T7  ,  5 $
*G7  ,*T8  ,  5 $
*G8  ,*T9  ,  5 $
*G9  ,*T10 ,  5 $
*G10 ,*T11 ,  5 $
*G11 ,*T13 ,  7 $
*G12 ,*T14 ,  7 $
*G13 ,*T15 ,  7 $
*G14 ,*T16 ,  7 $
*G15 ,*T17 ,  7 $
*G16 ,*T19 ,  8 $
*G17 ,*T20 ,  8 $
*G18 ,*T21 ,  8 $
*G19 ,*T22 ,  8 $
*G20 ,*T23 ,  8 $
*G21 ,*T25 , 24 $
*G22 ,*T26 , 24 $
*G23 ,*T27 , 24 $
*G24 ,*T28 , 24 $
*G25 ,*T29 , 24 $
*G26 ,*T31 , 23 $
*G27 ,*T32 , 23 $
*G28 ,*T33 , 23 $
*G29 ,*T34 , 23 $
*G30 ,*T35 , 23 $
*G31 ,*T37 , 21 $
*G32 ,*T38 , 21 $
*G33 ,*T39 , 21 $
*G34 ,*T40 , 21 $
*G35 ,*T41 , 21 $
*G36 ,*T43 , 22 $
*G37 ,*T44 , 22 $
*G38 ,*T45 , 22 $
*G39 ,*T46 , 22 $
*G40 ,*T47 , 22 $
```

ADAT array (cont'd)

ADAT1 array described on page 53 and the third argument to FLOCN1 on page 121

```
*G41 ,*T49 , 16 $
*G42 ,*T50 , 16 $
*G43 ,*T51 , 16 $
*G44 ,*T52 , 16 $
*G45 ,*T53 , 16 $
*G46 ,*T55 , 15 $
*G47 ,*T56 , 15 $
*G48 ,*T57 , 15 $
*G49 ,*T58 , 15 $
*G50 ,*T59 , 15 $
*G51 ,*T61 , 13 $
*G52 ,*T62 , 13 $
*G53 ,*T63 , 13 $
*G54 ,*T64 , 13 $
*G55 ,*T65 , 13 $
*G56 ,*T67 , 14 $
*G57 ,*T68 , 14 $
*G58 ,*T69 , 14 $
*G59 ,*T70 , 14 $
*G60 ,*T71 , 14 $
*G61 ,*T73 , 30 $
*G62 ,*T74 , 30 $
*G63 ,*T75 , 30 $
*G64 ,*T76 , 30 $
*G65 ,*T77 , 30 $
*G66 ,*T79 , 29 $
*G67 ,*T80 , 29 $
*G68 ,*T81 , 29 $
*G69 ,*T82 , 29 $
*G70 ,*T83 , 29 $
*G71 ,*T85 , 31 $
*G72 ,*T86 , 31 $
*G73 ,*T87 , 31 $
*G74 ,*T88 , 31 $
*G75 ,*T89 , 31 $
*G76 ,*T91 , 32 $
*G77 ,*T92 , 32 $
*G78 ,*T93 , 32 $
*G79 ,*T94 , 32 $
*G80 ,*T95 , 32 $
*G81 ,*T200,  1 $
*G82 ,*T97 ,  2 $
*G83 ,*T97 , 36 $
*G84 ,*T98 ,  3 $
*G85 ,*T98 , 11 $
*G86 ,*T99 ,  7 $
*G87 ,*T99 ,  8 $
*G88 ,*T99 ,  4 $
*G89 ,*T100,  6 $
*G90 ,*T100,  5 $
*G91 ,*T18 ,  7 $
*G92 ,*T24 ,  8 $
*G93 ,*T 6 ,  6 $
*G94 ,*T12 ,  5 $
*G95 ,*T101,  9 $
*G96 ,*T102, 19 $
*G97 ,*T103, 24 $
*G98 ,*T103, 23 $
```

ADAT1 array (cont'd)

```
*G99 ,*T103, 20 $
*G100,*T104, 21 $
*G101,*T104, 22 $
*G102,*T42 , 21 $
*G103,*T48 , 22 $
*G104,*T105, 25 $
*G105,*T107, 16 $
*G106,*T107, 15 $
*G107,*T107, 12 $
*G108,*T108, 13 $
*G109,*T108, 14 $
*G110,*T54 , 16 $
*G111,*T60 , 15 $
*G112,*T66 , 13 $
*G113,*T72 , 14 $
*G114,*T109, 17 $
*G115,*T110, 27 $
*G116,*T111, 31 $
*G117,*T111, 32 $
*G118,*T111, 28 $
*G119,*T112, 30 $
*G120,*T112, 29 $
*G121,*T90 , 31 $
*G122,*T96 , 32 $
*G123,*T78 , 30 $
*G124,*T84 , 29 $
*G125,*T113, 33 $
*G126,*T106, 26 $
*G127,*T114, 34 $
*G128,*T115, 35 $
*G129,*T116, 36 $
*G130,*T30 , 24 $
*G131,*T36 , 23 $
```

ADAT1 array (cont'd)

```
END
  20   $ AFLOW
    *A21  $ ARRAY CONTAINING FLOW RATES
    *A22  $ ARRAY CONTAINING PRESSURES
    *A23  $ ARRAY CONTAINING FLOW CONDUCTORS
    *A24  $ ARRAY CONTAINING VALVE POSITIONS
    *A25  $ ARRAY CONTAINING P-NODE IMPOSED FLOW RATES
    *A26  $ ARRAY CONTAINING FLUID LUMP TYPE DATA
    *A27  $ ARRAY CONTAINING ADDED RESISTANCES
    *A28  $ ARRAY CONTAINING PRESSURE DROPS
  END
```

AFLOW array described on page 71 and the first argument in the call to PFCS on page 121 (also first argument to CONV1)

```
  21   $ FLOW RATES ,AW
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.   ,   200.  ,   200.  ,  200.
  200.  ,  200.   ,  200.
```

AW array described on page 72 and referenced in array 20 above

```
END
  22   $ PRESSURES
   SPACE,24
```

APN array described on page 72

```
END
  23   $ FLOW CONDUCTORS
   SPACE,39
```

AGF array described on page 72

```
END
  24    $ VALVE POSITIONS
    0.999, 0.99
END
  25    $ IMPOSED FLOW RATES
            200.,SPACE,23
```

AVP array described on page 72

AIFR array described on page 72

```
END
  26   $ FLUID TYPE DATA--WP,CSA,FLL,MFF,NHL,FFC
REM  WP      , CSA      , FLL     , MFF ,NHL, FFC $
   .1125  , .001008 ,  12.     ,   0, 0. , 1.  $ TYPE 1
   .3600  , .000938 ,   3.25   ,   0,117., 1.  $ TYPE 2
   .1125  , .001008 ,   5.     ,   0, 0. , 1.  $ TYPE 3
   .0328  , .853E-4 ,   0.25   ,   0,2.49, 1.  $ TYPE 4
   .1125  , .001008 ,  20.     ,   0, 0. , 1.  $ TYPE 5
   .1125  , .001008 ,   2.5    ,   0, 0. , 1.  $ TYPE 6
   .1125  , .001008 ,  50.     ,   0, 0. , 1.  $ TYPE 7
   .1125  , .001008 ,   7.     ,   0, 0. , 1.  $ TYPE 8
   .1125  , .001008 ,   2.     ,   0, 0. , 1.  $ TYPE 9
```

AFT array containing the fluid type data described on page 72 and referenced in array 51.

```
END
  27   $ ADDED RESISTANCES
   SPACE, 39
```

AFR array described on page 72

```
END
  28   $ PRESSURE DROPS
   SPACE, 39
```

APD array described on page 73

```
END
  30    $   SYSTEM ARRAYS,ADAT
    *A31   $ ID ARRAY CONTAINING SYSTEM PROPERTY ID
    *A32   $ ID ARRAY CONTAINING SOLUTION PARAMETERS
    *A33   $ ID ARRAY CONTAINING MAIN NETWORK
    *A36   $ ID ARRAY CONTAINING ID OF VALVE DATA
    *A35   $ ID ARRAY CONTAINING PUMP DATA
       0   $ CHECKOUT PRINT CODE
```

ADAT array described on page 71 and the second argument in the call to PFCS on page 121

```
  END
  31
    *A1    $ CP ARRAY
    *A2    $ RO ARRAY
    *A5    $ MU ARRAY
    *A6    $ KT ARRAY
 417312000.$ GC
```

APR array described on page 73 and identified in array 30

```
   END
  32       $ ASOL,SOLUTION PARAMETERS
    0.01, 100, 0.0, 0.7  $ TOL, MXPASS, EPS, FRCF
```

ASOL array described on page 74 and identified in array 30

```
 END
  33
    MAIN       $ NAME
    *A34   $ ARRAY ID PRESS NODES W/P SPECIFIED
    *A36   $ ARRAY ID ARRAYS CONTAINING VALVE DATA
REM    TUBE ,FROM,TO   ,SS,FL+TR $
REM         ,P-NO,P-NO,KO,LUMP A$
          1,   1,   2, 1,*A37 $ MAIN NETWORK
          2,   2,   3, 1,*A38
          3,   3,   4, 1,*A39
         38,   4,  17,-1,*A40 $ SUBNETWORK 1
         26,  17,  18, 1,*A41
         11,   3,   9, 1,*A42
         39,   9,  22,-1,*A43 $ SUBNETWORK 2
         34,  22,  18, 1,*A44
         35,  18,  23, 1,*A45
```

ANET array for the main network described on page 74 and identified in array 30.

```
      36,   2,  23, 1,*A46
      37,  23,  24, 1,*A47
  END
    34, 24,END   $ NODES W/SPECIFIED PRESSURES
    35,  1, 24, *A13, END
    36,    *A48,*A49,END  $ ARRAYS CONTAINING VALVE DATA
REM AND ,FLUID,TYP,TUBE
REM      ,LUMP ,    ,LUMP
    37,*T97 ,  3,*T297 ,END    $ TUBE 1
    38,*T98 ,  3,*T298 ,END    $ TUBE 2
    39,*T99 ,  5,*T299 ,END    $ TUBE 3
    41,*T106,  5,*T306 ,END    $ TUBE 26
    42,*T107,  5,*T307 ,END    $ TUBE 11
    44,*T114,  5,*T314 ,END    $ TUBE 34
    45,*T115,  7,*T315 ,END    $ TUBE 35
    46,*T116,  9,*T316 ,END    $ TUBE 36
    47,*T117,  3,*T317 ,END    $ TUBE 37
    48,1,2,36,1,.001,.999,.01,*T117,35.,.75,1.,10.
    END     $ DATA FOR VALVE 1
    49, 2, 3,11, 1,  0.01, 0.99, 0.01,*T115, 40.,0.75,1.0,10.
    END     $ DATA FOR VALVE 2
    40
      SUB1  $  NAME
         0  $
         0  $
REM    TUBE ,FROM,TO   ,SS,FL+TB $
REM         ,P-NO,P-NO,NO,LUMP A$
          4,   4,   6, 1,  *A51  $ SUBNETWORK 1
          5,   6,   7, 1,  *A52  $
          6,   6,   7, 1,  *A53  $
          7,   4,   5, 1,  *A54  $
          8,   4,   5, 1,  *A55  $
          9,   5,   7, 1,  *A56  $
         10,   7,   8, 1,  *A57  $
         19,   8,  14, 1,  *A58  $
         20,  14,  16, 1,  *A59  $
         21,  16,  17, 1,  *A60  $
         22,  16,  17, 1,  *A61  $
         23,  14,  15, 1,  *A62  $
         24,  14,  15, 1,  *A63  $
         25,  15,  17, 1,  *A64  $
    END                          $
REM AND ,FLUID,TYP,TUBE
REM      ,LUMP ,    ,LUMP
    51, *T105, 8, *T305, END    $ TUBE  4
    52, *T7  , 3, *T207         $ TUBE  5
        *T8  , 2, *T208         $
        *T9  , 2, *T209         $
        *T10 , 2, *T210         $
        *T11 , 2, *T211         $
        *T12 , 3, *T212, END    $
    53, *T1  , 1, *T201         $ TUBE  6
        *T2  , 2, *T202         $
        *T3  , 2, *T203         $
        *T4  , 2, *T204         $
        *T5  , 2, *T205         $
        *T6  , 1, *T206, END    $
    54, *T13 , 3, *T213         $ TUBE  7
```

(Lines 36–37: array 33)

(Line 34) — APNPS array described on p. 75 & identified in/ array 33

(Line 35) — AP, array containing pump data described on page 77 and identified in array 30

(Line 36) — AVLS and AVL array described on page 76 and referenced in array 30(AVLS) and array 33(AVL)

(Arrays 37–47) ADi arrays for the main network; described on page 75 and referenced in array 33

(Arrays 48–49) AVLV arrays containing valve data described on page 76 and referenced in the AVL array 36

(Array 40) ANET array for subnetwork 1; described on page 74 and identified in ANET for the main network, array 33

(Arrays 51–54) ADi arrays for subnetwork No. 1 described on page 75 and referenced in array 40

```
        *T14 , 2, *T214          $
        *T15 , 2, *T215          $
        *T16 , 2, *T216          $
        *T17 , 2, *T217          $
        *T18 , 3, *T218, END     $
    55, *T19 , 1, *T219          $ TUBE  8
        *T20 , 2, *T220          $
        *T21 , 2, *T221          $
        *T22 , 2, *T222          $
        *T23 , 2, *T223          $
        *T24 , 1, *T224, END     $
    56, *T101, 8, *T301, END     $ TUBE  9
    57, *T102, 6, *T302, END     $ TUBE 10
    58, *T103, 6, *T303, END     $ TUBE 19
    59, *T104, 8, *T304, END     $ TUBE 20
    60, *T37 , 3, *T237          $ TUBE 21
        *T38 , 2, *T238          $
        *T39 , 2, *T239          $
        *T40 , 2, *T240          $
        *T41 , 2, *T241          $
        *T42 , 3, *T242, END     $
    61, *T43 , 1, *T243          $ TUBE 22
        *T44 , 2, *T244          $
        *T45 , 2, *T245          $
        *T46 , 2, *T246          $
        *T47 , 2, *T247          $
        *T48 , 1, *T248, END     $
    62, *T31 , 3, *T231          $ TUBE 23
        *T32 , 2, *T232          $
        *T33 , 2, *T233          $
        *T34 , 2, *T234          $
        *T35 , 2, *T235          $
        *T36 , 3, *T236, END     $
    63, *T25 , 1, *T225          $ TUBE 24
        *T26 , 2, *T226          $
        *T27 , 2, *T227          $
        *T28 , 2, *T228          $
        *T29 , 2, *T229          $
        *T30 , 1, *T230, END     $
    64, *T105, 8, *Y305, END     $ TUBE 25
   93
      SUB2  $  NAME
         0  $
         0  $
REM    TUBE ,FROM,TO   ,SS,FL+TB $
REM         ,P-ND,P-ND,KJ,LUMP AS
         12,    9,   11, 1,   *A71  $ SUBNETWORK 1
         13,   11,   12, 1,   *A72  $
         14,   11,   12, 1,   *A73  $
         15,    9,   10, 1,   *A74  $
         16,    9,   10, 1,   *A75  $
         17,   10,   12, 1,   *A76  $
         18,   12,   13, 1,   *A77  $
         27,   13,   19, 1,   *A78  $
         28,   19,   21, 1,   *A79  $
         29,   21,   22, 1,   *A80  $
         30,   21,   22, 1,   *A81  $
         31,   19,   20, 1,   *A82  $
```

ADi arrays for subnetwork No. 1 (Cont'd)

ANET array for subnetwork 2; described on page and identified in ANET for the main network, array 33

```
        32,  19,  20, 1,  *A83  $
        33,  20,  22, 1,  *A84  $
     END                        $
REM AND ,FLUID,TYP,TUBE
REM     ,LUMP ,    ,LUMP
     71, *T108, 8, *T308, END   $ TUBE 12
     72, *T61 , 3, *T261        $ TUBE 13
         *T62 , 4, *T262        $
         *T63 , 4, *T263        $
         *T64 , 4, *T264        $
         *T65 , 4, *T265        $
         *T66 , 3, *T266, END   $
     73, *T67 , 1, *T267        $ TUBE 14
         *T68 , 4, *T268        $
         *T69 , 4, *T269        $
         *T70 , 4, *T270        $
         *T71 , 4, *T271        $
         *T72 , 1, *T272, END   $
     74, *T55 , 3, *T255        $ TUBE 15
         *T56 , 4, *T256        $
         *T57 , 4, *T257        $
         *T58 , 4, *T258        $
         *T59 , 4, *T259        $
         *T60 , 3, *T260, END   $
     75, *T49 , 1, *T249        $ TUBE 16
         *T50 , 4, *T250        $
         *T51 , 4, *T251        $
         *T52 , 4, *T252        $
         *T53 , 4, *T253        $
         *T54 , 1, *T254, END   $
     76, *T109, 8, *T309, END   $ TUBE 17
     77, *T110, 6, *T310, END   $ TUBE 18
     78, *T111, 6, *T311, END   $ TUBE 27
     79, *T112, 8, *T312, END   $ TUBE 28
     80, *T79 , 3, *T279        $ TUBE 29
         *T80 , 4, *T280        $
         *T81 , 4, *T281        $
         *T82 , 4, *T282        $
         *T83 , 4, *T283        $
         *T84 , 3, *T284, END   $
     81, *T73 , 1, *T273        $ TUBE 30
         *T74 , 4, *T274        $
         *T75 , 4, *T275        $
         *T76 , 4, *T276        $
         *T77 , 4, *T277        $
         *T78 , 1, *T278, END   $
     82, *T85 , 3, *T285        $ TUBE 31
         *T86 , 4, *T286        $
         *T87 , 4, *T287        $
         *T88 , 4, *T288        $
         *T89 , 4, *T289        $
         *T90 , 3, *T290, END   $
     83, *T91 , 1, *T291        $ TUBE 32
         *T92 , 4, *T292        $
         *T93 , 4, *T293        $
         *T94 , 4, *T294        $
         *T95 , 4, *T295        $
         *T96 , 1, *T296, END   $
```

ADi array for subnetwork No. 2 described on page 75 and referenced in array 43

```
        84, +T113, 8, +T313, END   $ TUBE 33
        100
  BCD 9SHUTTLE ECS RADIATOR FLOW SYSTEM
        END
        110
  BCD 9FLOW RATES (LB/HR)
        END
        111
  BCD 9PRESSURES  (LB/FT**2)
        END
        112
  BCD 9VALVE POSITIONS
        END
        113
  BCD 9PRESSURE DROPS  (LB/FT**2)
       END
  END
  ARRAY ANALYSIS...  NUMBER OF ARRAYS =   75  TOTAL LENGTH =  2511

  BCD 3EXECUTION
 DIMENSION X(2000)                                                F
 NDIM = 2000                                                      F
 NTH  = 0                                                         F
       PFCS(A20,A30,A100+1)
       CNBACK
  END
  BCD 3VARIABLES 1
       DIDEG1(TIMEN,A11,T198)
       HXEFF (0.9,500.,A21+37, 1.0, A1, T198, T117, T199, T200)
       FLOCK1(A21,A1,A17)
       CONV1(A20,A31,A16)
  END
  BCD 3VARIABLES 2
       PFCS(A20,A30,A100+1)
       TIMCHK(K1,0)
       HSTRY(A22,A24,A21,DTIMEU)
  END
  BCD 3OUTPUT CALLS
       TPRINT
       FLPRNT(A21,A110+1)
       FLPRNT(A28,A113+1)
       FLPRNT(A22,A111+1)
       FLPRNT(A24,A112+1)
       TIMCHK(K1,1)
  END
```

```
                                                              25 FEB 73        14:15:36
@ XQT CUR                                                                      14:15:37
   1.  ERS                                                                     14:15:37
   2.  IN 6

     END OF FILE -- UNIT 6                                                     14:16: 7
   3.  TRI 6

     END CUR LCC 1102-0039 L9
```

```
@ FOR,X SINDA
UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A -(EXEC8 LEVEL E12010010A)
THIS COMPILATION WAS DONE ON 25 FEB 73 AT 14:16:07
```

MAIN PROGRAM

STORAGE USED: CODE(1) 000014; DATA(0) 000001; BLANK COMMON(2) 000000

COMMON BLOCKS:

0003	TITLE	000024
0004	TEMP	000356
0005	CAP	000352
0006	SOURCE	000352
0007	COND	000470
0010	PC1	000655
0011	PC2	000452
0012	KONST	000203
0013	ARRAY	004717
0014	FIXCON	000062
0015	DIMENS	000011
0016	LOGIC	000004

EXTERNAL REFERENCES (BLOCK, NAME)

0017	INPUTT
0020	EXECTN
0021	NSTOP$

STORAGE ASSIGNMENT (BLOCK, TYPE, RELATIVE LOCATION, NAME)

0001	000005	IL	0013	000000	A	0014	000022	ARLXCA	0014	000035	ARLXCC	0014	000012	ATMPCA
0014	000017	ATMPCC	0014	000013	BACKUP	0014	000040	BALENG	0005	000000	C	0014	000003	CSGFAC
0014	000026	CSGMAX	0014	000020	CSGMIN	0014	000027	CSGRAL	0014	000030	CSGRCL	0014	000010	DAMPA
0014	000011	DAMPD	0014	000031	DRLXCA	0014	000032	DRLXCC	0014	000007	DTIMEH	0014	000025	DTIMEI
0014	000024	DTIMEL	0014	000001	DTIMEU	0014	000005	DTMPCA	0014	000016	DTMPCC	0014	000037	ENGBAL
0007	000000	G	0003	000000	H	0014	000046	ITEST	0014	000047	JTEST	0012	000000	K
0014	000050	KTEST	0016 L	000003	LARRAY	0014	000060	LAXFAC	0016 L	000001	LCOND	0016 L	000002	LCONST
0015	000010	LENA	0014	000033	LINECT	0016 L	000000	LNODE	0014	000023	LOOPCT	0014	000036	LSPCS
0015	000006	LSQ1	0015	000007	LSQ2	0014	000051	LTEST	0014	000052	MTEST	0014	000044	NARLXC
0015	000005	NAT	0014	000045	NATMPC	0014	000042	NCSGM	0015	000004	NCT	0014	000043	NDTMPC
0015	000003	NGT	0014	000004	NLOOP	0015	000001	NNA	0015	000000	NND	0015	000002	NNT
0014	000041	NOCOPY	0014	000006	OPEITR	0014	000021	OUTPUT	0014	000034	PAGECT	0006	000000	Q
0014	000053	RTEST	0010	000000	SEQ1	0011	000000	SEQ2	0014	000054	STEST	0004	000000	T
0014	000061	TCNTAL	0014	000015	TIMEM	0014	000000	TIMEN	0014	000002	TIMEND	0014	000014	TIMEO
0014	000055	TTEST	0014	000056	UTEST	0014	000057	VTEST						

```
00101      1*         COMMON /TITLE/  H
00103      2*         COMMON /TEMP/   T
00104      3*         COMMON /CAP/    C
00105      4*         COMMON /SOURCE/ Q
00106      5*         COMMON /COND/   G
00107      6*         COMMON /PC1/  SEQ1
```

```
      COMMON /PC2/  SEQ2
      COMMON /KONST/  K
      COMMON /ARRAY/  A
      COMMON /FIXCON/   TIMEN ,     DTIMEU,     TIMEND,     CSGFAC,
     1NLOOP ,     DTMPCA,     OPEITR,     DTIMEH,     DAMPA ,
     1DAMPD ,     ATMPCA,     BACKUP,     TIMEO ,     TIMEM ,
     1DTMPCC,     ATMPCC,     CSGMIN,     OUTPUT,     ARLXCA,
     1LOOPCT,     DTIMEL,     DTIMEI,     CSGMAX,     CSGRAL,
     1CSGRCL,     DRLXCA,     DRLXCC,     LINECT,     PAGECT,
     1ARLXCC,     LSPCS ,     ENGBAL,     BALENG,     NOCOPY,
     1NCSGM ,     NDTMPC,     NARLXC,     NATMPC,     ITEST ,
     1JTEST ,     KTEST ,     LTEST ,     MTEST ,     RTEST ,
     1STEST ,     TTEST ,     UTEST ,     VTEST ,     LAXFAC,
     1TCNTRL
      COMMON /DIMENS/ NND,NNA,NNT,NGT,NCT,NAT,LSQ1,LSQ2,LENA
      DIMENSION  H(20)
      COMMON /LOGIC/  LNODE, LCOND, LCONST, LARRAY
      LOGICAL LNODE, LCOND, LCONST, LARRAY
      DIMENSION T(  238),C(  234),Q(  234),G(  312),K(  131),A( 2511),
     1SEQ1(  429),SEQ2(  298)
      LNODE  = .TRUE.
      LCOND  = .TRUE.
      LCONST = .TRUE.
      LARRAY = .TRUE.
    1 CALL  INPUTT
      CALL  EXECTN
      GO TO 1
      END
```

END OF COMPILATION: NO DIAGNOSTICS.

```
@ FOR,K EXECTN
UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A  -(EXEC8 LEVEL E12010010A)
THIS COMPILATION WAS DONE ON 25 FEB 73 AT 14:16:08

   SUBROUTINE EXECTN     ENTRY POINT 000021

   STORAGE USED: CODE(1) 000023; DATA(0) 000005; BLANK COMMON(2) 000000

    COMMON BLOCKS:

    0003   TITLE  000024
    0004   TEMP   000001
    0005   CAP    000001
    0006   SOURCE 000001
    0007   COND   000001
    0010   PC1    000001
    0011   PC2    000001
    0012   KONST  000001
    0013   ARRAY  000001
    0014   FIXCON 000062
    0015   DIMENS 000011
    0016   XSPACE 003722

   EXTERNAL REFERENCES (BLOCK, NAME)

    0017   PFCS
    0020   CNBACK
    0021   NERR3$

   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)
```

0013 R	000000 A	0014	000022 ARLXCA	0014	000035 ARLXCC	0014	000012 ATMPCA	0014	000017 ATMPCC
0014	000013 BACKUP	0014	000040 BALENG	0005	000000 C	0014	000003 CSGFAC	0014	000026 CSGMAX
0014	000020 CSGMIN	0014	000027 CSGRAL	0014	000030 CSGRCL	0014	000010 DAMPA	0014	000011 DAMPD
0014	000031 DRLXCA	0014	000032 DRLXCC	0014	000007 DTIMEH	0014	000025 DTIMEI	0014	000024 DTIMEL
0014	000001 DTIMEU	0014	000005 DTMPCA	0014	000016 DTMPCC	0014	000037 ENGBAL	0007	000000 G
0003	000000 H	0000	000000 INJP$	0014	000046 ITEST	0014	000047 JTEST	0012	000000 K
0014	000050 KTEST	0014	000060 LAXFAC	0015	000010 LENA	0014	000033 LINECT	0014	000023 LOOPCT
0014	000036 LSPCS	0015	000006 LSQ1	0015	000007 LSQ2	0014	000051 LTEST	0014	000052 MTEST
0014	000044 NARLXC	0015	000005 NAT	0014	000045 NATMPC	0014	000042 NCSGM	0015	000004 NCT
0016 I	000000 NDIM	0014	000043 NDTMPC	0015	000003 NGT	0014	000004 NLOOP	0015	000001 NNA
0015	000000 NND	0015	000002 NNT	0014	000041 NOCOPY	0016 I	000001 NTH	0016	000002 NX
0014	000006 OPEITR	0014	000021 OUTPUT	0014	000034 PAGECT	0006	000000 Q	0014	000053 RTEST
0010	000000 SEQ1	0011	000000 SEQ2	0014	000054 STEST	0004	000000 T	0014	000061 TCNTRL
0014	000015 TIMEM	0014	000000 TIMEN	0014	000002 TIMEND	0014	000014 TIMEO	0014	000055 TTEST
0014	000056 UTEST	0014	000057 VTEST	0016	000002 X	0012	000000 XK		

```
00101      1*           SUBROUTINE  EXECTN
00103      2*           COMMON /TITLE/  H
00104      3*           COMMON /TEMP/   T
00105      4*           COMMON /CAP/    C
```

```
00106          COMMON /SOURCE/ Q
00107          COMMON /COND/   G
00110          COMMON /PC1/  SEQ1
00111          COMMON /PC2/  SEQ2
00112          COMMON /KONST/  K
00113          COMMON /ARRAY/  A
00114          COMMON /FIXCON/   TIMEN ,      DTIMEU,      TIMEND,      CSGFAC,
00114         1NLOOP ,      DTMPCA,      OPEITR,      DTIMEH,      DAMPA ,
00114         1DAMPD ,      ATMPCA,      BACKUP,      TIMEO ,      TIMEM ,
00114         1DTMPCC,      ATMPCC,      CSGMIN,      OUTPUT,      ARLXCA,
00114         1LOOPCT,      DTIMEL,      DTIMEI,      CSGMAX,      CSGRAL,
00114         1CSGRCL,      DRLXCA,      DRLXCC,      LINECT,      PAGECT,
00114         1ARLXCC,      LSPCS ,      ENGBAL,      BALENG,      NOCOPY,
00114         1NCSGM ,      NDTMPC,      NARLXC,      NATMPC,      ITEST ,
00114         1JTEST ,      KTEST ,      LTEST ,      MTEST ,      RTEST ,
00114         1STEST ,      TTEST ,      UTEST ,      VTEST ,      LAXFAC,
00114         1TCNTRL
00115          COMMON /DIMENS/ NND,NNA,NNT,NGT,NCT,NAT,LSQ1,LSQ2,LENA
00116          DIMENSION  H(20)
00117          COMMON /XSPACE/  NDIM, NTH, X
00120          DIMENSION   T(1), C(1), Q(1), G(1), K(1), A(1)
00121          DIMENSION IK(1), NX(1)
00122          EQUIVALENCE (K,IK), (X,NX)
00123          DIMENSION X(2000)
00124          NDIM = 2000
00125          NTH  = 0
00126          CALL  PFCS  (A(1537),A(1814),A(2463))
00127          CALL  CNBACK
00130          RETURN
00131          END
```

END OF COMPILATION: NO DIAGNOSTICS.

```
@ FOR,X VARBL1
UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A  -(EXEC8 LEVEL E12010010A)
THIS COMPILATION WAS DONE ON 25 FEB 73 AT 14:16:09

  SUBROUTINE VARBL1    ENTRY POINT 000041

  STORAGE USED: CODE(1) 000043; DATA(0) 000010; BLANK COMMON(2) 000000

   COMMON BLOCKS:

   0003   TITLE  000024
   0004   TEMP   000001
   0005   CAP    000001
   0006   SOURCE 000001
   0007   COND   000001
   0010   PC1    000001
   0011   PC2    000001
   0012   KONST  000001
   0013   ARRAY  000001
   0014   FIXCON 000062
   0015   DIMENS 000011
   0016   XSPACE 000003

  EXTERNAL REFERENCES (BLOCK, NAME)

   0017   D1DEG1
   0020   HXEFF
   0021   FLOCN1
   0022   CONV1
   0023   NERR3$

  STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

   0013 R 000000 A        0014   000022 ARLXCA   0014   000035 ARLXCC   0014   000012 ATMPCA   0014   000017 ATMPCC
   0014   000013 BACKUP   0014   000040 BALENG   0005   000000 C        0014   000003 CSGFAC   0014   000026 CSGMAX
   0014   000020 CSGMIN   0014   000027 CSGRAL   0014   000030 CSGRCL   0014   000010 DAMPA    0014   000011 DAMPC
   0014   000031 DRLXCA   0014   000032 DRLXCC   0014   000007 DTIMEH   0014   000025 DTIMEI   0014   000024 DTIMEL
   0014   000001 DTIMEU   0014   000005 DTMPCA   0014   000016 DTMPCC   0014   000037 ENGBAL   0007   000000 G
   0003   000000 H        0000   000003 INJP$    0014   000046 ITEST    0014   000047 JTEST    0012   000000 K
   0014   000050 KTEST    0014   000060 LAXFAC   0015   000010 LENA     0014   000033 LINECT   0014   000023 LOOPCT
   0014   000036 LSPCS    0015   000006 LSQ1     0015   000007 LSQ2     0014   000051 LTEST    0014   000052 MTEST
   0014   000044 NARLXC   0015   000005 NAT      0014   000045 NATMPC   0014   000042 NCSGM    0015   000004 NCT
   0016   000000 NDIM     0014   000043 NDTMPC   0015   000003 NGT      0014   000004 NLOOP    0015   000001 NNA
   0015   000000 NND      0015   000002 NNT      0014   000041 NOCOPY   0014   000002 NTH      0016   000002 NX
   0014   000006 OPEITR   0014   000021 OUTPUT   0014   000034 PAGECT   0006   000000 Q        0014   000053 RTEST
   0010   000000 SEQ1     0011   000000 SEQ2     0014   000054 STEST    0004 R 000000 T        0014   000061 [illegible]
   0014   000015 TIMEM    0014 R 000000 TIMEN    0014   000002 TIMEND   0014   000014 TIMEO    0014   000055 TTEST
   0014   000056 UTEST    0014   000057 VTEST    0016   000002 X        0012   000000 XK

00101         1*            SUBROUTINE  VARBL1
00103         2*            COMMON /TITLE/  H
```

```
00104        COMMON /TEMP/   T
00105        COMMON /CAP/    C
00106        COMMON /SOURCE/ Q
00107        COMMON /COND/   G
00110        COMMON /PC1/  SEQ1
00111        COMMON /PC2/  SEQ2
00112        COMMON /KONST/  K
00113        COMMON /ARRAY/  A
00114        COMMON /FIXCON/   TIMEN ,     DTIMEU,     TIMEND,     CSGFAC,
00114       1NLOOP ,     DTMPCA,     OPEITR,     DTIMEH,     DAMPA ,
00114       1DAMPD ,     ATMPCA,     BACKUP,     TIMEO ,     TIMEM ,
00114       1DTMPCC,     ATMPCC,     CSGMIN,     OUTPUT,     ARLXCA,
00114       1LOOPCT,     DTIMEL,     DTIMEI,     CSGMAX,     CSGRAL,
00114       1CSGRCL,     DRLXCA,     DRLXCC,     LINECT,     PAGECT,
00114       1ARLXCC,     LSPCS ,     ENGBAL,     BALENG,     NOCOPY,
00114       1NCSGM ,     NDTMPC,     NARLXC,     NATMPC,     ITEST ,
00114       1JTEST ,     KTEST ,     LTEST ,     MTEST ,     RTEST ,
00114       1STEST ,     TTEST ,     UTEST ,     VTEST ,     LAXFAC,
00114       1TCNTRL
00115        COMMON /DIMENS/ NND,NNA,NNT,NGT,NCT,NAT,LSQ1,LSQ2,LENA
00116        DIMENSION  H(20)
00117        COMMON /XSPACE/  NDIM, NTH, X
00120        DIMENSION   T(1), C(1), Q(1), G(1), K(1), A(1)
00121        DIMENSION XK(1), NX(1)  , X(1)
00122        EQUIVALENCE (K,XK), (X,NX)
00123        CALL  DIDEGI(TIMEN ,A(190),T(235))
00124        CALL  HXEFF (0.9,500.,A(1583),1.0,A(1),T(235),T(117),
00124       2T(236),T(237))
00125        CALL  FLOCNI(A(1596),A(1),A(1143))
00126        CALL  CONVI (A(1537),A(1821),A(214))
00127        RETURN
00130        END

END OF COMPILATION:        NO  DIAGNOSTICS.
```

```
@ FOR,K VARBL2
UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A  -(EXEC8 LEVEL E1201001OA)
THIS COMPILATION WAS DONE ON 25 FEB 73 AT 14:16:10


   SUBROUTINE VARBL2    ENTRY POINT 000026


   STORAGE USED: CODE(1) 000030; DATA(0) 000006; BLANK COMMON(2) 000000

    COMMON BLOCKS:

    0003   TITLE  000024
    0004   TEMP   000001
    0005   CAP    000001
    0006   SOURCE 000001
    0007   COND   000001
    0010   PC1    000001
    0011   PC2    000001
    0012   KONST  000001
    0013   ARRAY  000001
    0014   FIXCON 000062
    0015   DIMENS 000011
    0016   XSPACE 000003


   EXTERNAL REFERENCES (BLOCK, NAME)

    0017   PFCS
    0020   TIMCHK
    0021   MSTRY
    0022   NERR3$


   STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0013 R 000000 A        0014   000022 ARLXCA    0014   000035 ARLXCC    0014   000012 ATMPCA    0014   000017 ATMPCC
    0014   000013 BACKUP   0014   000040 BALENG    0005   000000 C         0014   000003 CSGFAC    0014   000026 CSGMAX
    0014   000020 CSGMIN   0014   000027 CSGRAL    0014   000030 CSGRCL    0014   000010 DAMPA     0014   000011 DAMPD
    0014   000031 DRLXCA   0014   000032 DRLXCC    0014   000007 DTIMEH    0014   000025 DTIMEI    0014   000024 DTIMEL
    0014 R 000001 DTIMEU   0014   000005 DTMPCA    0014   000016 DTMPCC    0014   000037 ENGBAL    0007   000000 G
    0003   000000 H        0000   000001 INJP$     0014   000046 ITEST     0014   000047 JTEST     0012 I 000000 K
    0014   000050 KTEST    0014   000060 LAXFAC    0015   000010 LENA      0014   000033 LINECT    0014   000023 LOOPCT
    0014   000036 LSPCS    0015   000006 LSQ1      0015   000007 LSQ2      0014   000051 LTEST     0014   000052 MTEST
    0014   000044 NARLXC   0015   000005 NAT       0014   000045 NATMPC    0014   000042 NOSCP     0015   000004 NCT
    0016   000000 NDIM     0014   000043 NDTMPC    0015   000003 NGT       0014   000004 NLOOP     0015   000001 NNA
    0015   000000 NND      0015   000002 NNT       0014   000041 NOCOPY    0016   000001 NTH       0016   000002 NX
    0014   000006 OPEITR   0014   000021 OUTPUT    0014   000034 PASECT    0006   000000 Q         0014   000053 RTEST
    0010   000000 SEQ1     0011   000000 SEQ2      0014   000054 STEST     0004   000000 T         0014   000061 TENTRL
    0014   000015 TIMEM    0014   000000 TIMEN     0014   000002 TIMEND    0014   000014 TIMEO     0014   000055 TTEST
    0014   000056 UTEST    0014   000057 VTEST     0016   000002 X         0012   000000 XK


 00101      1*          SUBROUTINE  VARBL2
 00103      2*          COMMON /TITLE/  H
 00104      3*          COMMON /TEMP/   T
```

```
      COMMON /CAP/    C
      COMMON /SOURCE/ Q
      COMMON /COND/   G
      COMMON /PC1/  SEQ1
      COMMON /PC2/  SEQ2
      COMMON /KONST/  K
      COMMON /ARRAY/  A
      COMMON /FIXCON/   TIMEN ,     DTIMEU,     TIMEND,     CSGFAC,
     1NLOOP ,     DTMPCA,     OPEITR,     DTIMEH,     DAMPA ,
     1DAMPD ,     ATMPCA,     BACKUP,     TIMEO ,     TIMEM ,
     1DTMPCC,     ATMPCC,     CSGMIN,     OUTPUT,     ARLXCA,
     1LOOPCT,     DTIMEL,     DTIMEI,     CSGMAX,     CSGRAL,
     1CSGRCL,     DRLXCA,     DRLXCC,     LINECT,     PAGECT,
     1ARLXCC,     LSPCS ,     ENGBAL,     BALENG,     NOCOPY,
     1NCSGM ,     NDTMPC,     NARLXC,     NATMPC,     ITEST ,
     1JTEST ,     KTEST ,     LTEST ,     MTEST ,     RTEST ,
     1STEST ,     TTEST ,     UTEST ,     VTEST ,     LAXFAC,
     1TCNTRL
      COMMON /DIMENS/ NND,NNA,NNT,NGT,NCT,NAT,LSQ1,LSQ2,LENA
      DIMENSION  H(20)
      COMMON /XSPACE/  NDIM, NTH, X
      DIMENSION    T(1), C(1), Q(1), G(1), K(1), A(1)
      DIMENSION XK(1), NX(1)   , X(1)
      EQUIVALENCE (K,XK), (X,NX)
      CALL  PFCS  (A(1537),A(1814),A(2463))
      CALL  TIMCHK(K(1),0)
      CALL  HSTRY (A(1586),A(1651),A(1596),DTIMEU)
      RETURN
      END
```

END OF COMPILATION: NO DIAGNOSTICS.

```
@ FOR,I OUTCAL
UNIVAC 1108 FORTRAN V EXEC II LEVEL 25A  -(EXEC8 LEVEL E1201001OA)
THIS COMPILATION WAS DONE ON 25 FEB 73 AT 19:16:12

 SUBROUTINE OUTCAL    ENTRY POINT 000035

 STORAGE USED: CODE(1) 000037; DATA(0) 000006; BLANK COMMON(2) 000000

  COMMON BLOCKS:

  0003   TITLE  000024
  0004   TEMP   000001
  0005   CAP    000001
  0006   SOURCE 000001
  0007   COND   000001
  0010   PC1    000001
  0011   PC2    000001
  0012   KONST  000001
  0013   ARRAY  000001
  0014   FIXCON 000062
  0015   DIMENS 000011
  0016   XSPACE 000003

 EXTERNAL REFERENCES (BLOCK, NAME)

  0017   TPRINT
  0020   FLPRNT
  0021   TIMCHK
  0022   NERR3$

 STORAGE ASSIGNMENT  (BLOCK, TYPE, RELATIVE LOCATION, NAME)

  0013 R 000000 A        0014   000022 AALXCA    0014   000035 AALXCC    0014   000012 ATMPCA    0014   000017 ATMPCC
  0014   000013 BACKUP   0014   000040 BALENG    0005   000000 C         0014   000003 CSGFAC    0014   000026 CSGMAX
  0014   000020 CSGMIN   0014   000027 CSGRAL    0014   000030 CSGRCL    0014   000010 DAMPA     0014   000011 DAMPD
  0014   000031 DRLXCA   0014   000032 DRLXCC    0014   000007 DTIMEH    0014   000025 DTIMEI    0014   000024 DTIMEL
  0014   000001 DTIMEU   0014   000005 DTMPCA    0014   000016 DTMPCC    0014   000037 ENGBAL    0007   000000 G
  0003   000000 H        0000   000001 INJP$     0014   000046 ITEST     0014   000047 JTEST     0012 I 000000 K
  0014   000050 KTEST    0014   000060 LAXFAC    0015   000010 LENA      0014   000033 LINECT    0014   000023 LOOPCT
  0014   000036 LSPCS    0015   000006 LSQ1      0015   000007 LSQ2      0014   000051 LTEST     0014   000052 MTEST
  0014   000044 NARLXC   0015   000005 NAT       0014   000045 NATMPC    0014   000042 NCSGM     0015   000004 NCT
  0016   000000 NDIM     0014   000043 NDTMPC    0015   000003 NGT       0014   000004 NLOOP     0015   000001 NNA
  0015   000000 NND      0015   000002 NNT       0014   000041 NOCOPY    0016   000001 NTH       0016   000002 NX
  0014   000006 OPEITR   0014   000021 OUTPUT    0014   000034 PAGECT    0006   000000 Q         0014   000053 RTEST
  0010   000000 SEQ1     0011   000000 SEQ2      0014   000054 STEST     0004   000000 T         0014   000061 TCNTRL
  0014   000015 TIMEM    0014   000000 TIMEN     0014   000002 TIMEND    0014   000014 TIMEO     0014   000055 TTEST
  0014   000056 UTEST    0014   000057 VTEST     0016   000002 X         0012   000000 XK


00101       1*          SUBROUTINE  OUTCAL
00103       2*          COMMON /TITLE/  H
00104       3*          COMMON /TEMP/   T
```

```
      COMMON /CAP/    C
      COMMON /SOURCE/ Q
      COMMON /COND/   G
      COMMON /PC1/  SEQ1
      COMMON /PC2/  SEQ2
      COMMON /KONST/  K
      COMMON /ARRAY/  A
      COMMON /FIXCON/   TIMEN ,     DTIMEU,     TIMEND,     CSGFAC,
     1NLOOP ,      DTMPCA,     DPEITR,     DTIMEH,     DAMPA ,
     1DAMPD ,      ATMPCA,     BACKUP,     TIMEO ,     TIMEM ,
     1DTMPCC,      ATMPCC,     CSGMIN,     OUTPUT,     ARLXCA,
     1LOOPCT,      DTIMEL,     DTIMEI,     CSGMAX,     CSGRAL,
     1CSGRCL,      DRLXCA,     DRLXCC,     LINECT,     PAGECT,
     1ARLXCC,      LSPCS ,     ENGBAL,     BALENG,     NOCOPY,
     1NCSGM ,      NDIMPC,     NARLXC,     NATMPC,     ITEST ,
     1JTEST ,      KTEST ,     LTEST ,     MTEST ,     RTEST ,
     1STEST ,      TTEST ,     UTEST ,     VTEST ,     LAXFAC,
     1TCNTRL
      COMMON /DIMENS/ NND,NNA,NNT,NGT,NCT,NAT,LSQ1,LSQ2,LENA
      DIMENSION  H(20)
      COMMON /XSPACE/  NDIM, NTH, X
      DIMENSION   T(1), C(1), Q(1), G(1), K(1), A(1)
      DIMENSION XK(1), NX(1)  , X(1)
      EQUIVALENCE (K,XK), (X,NX)
      CALL  TPRINT
      CALL  FLPRNT(A(1546),A(2473))
      CALL  FLPRNT(A(1774),A(2503))
      CALL  FLPRNT(A(1586),A(2483))
      CALL  FLPRNT(A(1651),A(2493))
      CALL  TIMCHK(K(1),1)
      RETURN
      END
```

END OF COMPILATION: NO DIAGNOSTICS.

@N HDG

```
STARTING ADDRESS  014000

CORE LIMITS 014000 044426     100000 123441     163772 163777


SINDA /CODE
 0   100000-100000
 1   014000-014013

NSTOP$/RLEC5
 1   014014-014025

NIER$ /RLEC5
 0   100001-100001
 1   014026-014333
 2   100002-100076

NFMT$ /RLEC4
 1   014334-015271
 2   100077-100113

NFTV$ /RL22
 1   015272-015314

NCNVT$/RLEC5
 1   015315-015541
 2   100114-100202

NOTIN$/RLEC5
 1   015542-016211
 2   100203-100246

FPACK$/CODE
 1   016212-016255

DEPTH /******
 0   100247-100254

NEAR$ /RLEC4
 0   100255-100444
 1   016256-016720

NIDIN$/RLEC5
 1   016721-016772
 2   100445-100475

EXECTN/CODE
 0   100476-100502
 1   016773-017015
```

```
NBACK/CODE
0   100503-100723
1   017016-023552

NOUTS /RLEC4
0   100724-100730
1   023553-024555
2   100731-100746

NTABS /CODE
0   100747-101115

NBDCVS/RLEC4
0   101116-101302

VARBL2/CODE
0   101303-101310
1   024556-024605

NSTRY /CODE
0   101311-101377
1   024606-025153

NRWNDS/RLEC4
1   025154-025246

NFOUTS/RLEC5
1   025247-025500
2   101400-101401

NBUFFS/RL23
1   025501-025523
2   101402-102412

NNREAD/CODE
0   102413-102452
1   025524-026226

NFINPS/RLEC5
1   026227-026470
2   102453-102453

TIMCHK/CODE
0   102454-102526
1   026471-026575

LINECK/CODE
0   102527-102534
1   026576-026631

CLOCK /CODE
0   102535-102537
1   026632-026712

TOPLIN/CODE
0   102540-102577
1   026713-026756

D2MXIM/CODE
0   102600-102614
1   026757-027101

D2CEGI/CODE
```

```
     102615-102667
1    027102-027456

DIDEGI/CODE
0    102670-102732
1    027457-027667

FLVARY/CODE
0    102733-102754
1    027670-030201

NEXP5S/AL24
1    030202-030265
2    102755-102764

DIMINM/CODE
0    102765-103003
1    030266-030451

OUTCAL/CODE
0    103004-103011
1    030452-030510

FLPRNT/CODE
0    103012-103043
1    030511-030705

TPRINT/CODE
0    103044-103133
1    030706-031210

STNDRD/CODE
0    103134-103176
1    031211-031336

VARBLI/CODE
0    103177-103206
1    031337-031401

CONV1 /CODE
0    103207-103505
1    031402-032263

NEXP6S/AL25
1    032264-032456
2    103506-103557

CBRT  /AL24
1    032457-032524
2    103560-103572

FPERR /CODE
0    103573-103574
1    032525-032535

FLDCNI/CODE
0    103575-103722
1    032536-033012

MXEFF /CODE
0    103723-104005
1    033013-033270
```

```
[illegible]CS  /CODE
0   104006-104657
1   033271-035306

EXP   /RL24
1   035307-035375
2   104660-104700

REVPOL/CODE
0   104701-104733
1   035376-035672

RTWRK /CODE
0   104734-105120
1   035673-036357

FLOBAL/CODE
0   105121-105545
1   036360-040070

SYMSOL/CODE
0   105546-105623
1   040071-040425

MFSD  /CODE
0   105624-105670
1   040426-040617

DSCRT /RL24
1   040620-040667
2   105671-105707

CMPRSS/CODE
0   105710-105752
1   040670-041140

GENOUT/CODE
0   105753-106027
1   041141-041535

PRN   /CODE
0   106030-106072
1   041536-041667

FLRES /CODE
0   106073-106253
1   041670-042321

SORT  /RL24
0   106254-106257
1   042322-042361
2   106260-106265

RTWRKI/CODE
0   106266-106451
1   042362-043054

RTWRKN/CODE
0   106452-106632
1   043055-043503

POINTN/******
0   106633-106637
```

```
SCATA /******
0   106640-106667

INPUTT/CODE
0   106670-106731
1   043504-044426

ISPACE/******
0   106732-112653

LOGIC /******
0   112654-112657

DIMENS/******
0   112660-112670

FIXCON/******
0   112671-112752

ARRAY /******
0   112753-117671

KONST /******
0   117672-120074

PC2   /******
0   120075-120546

PC1   /******
0   120547-121423

COND  /******
0   121424-122113

SOURCE/******
0   122114-122465

CAP   /******
0   122466-123037

TEMP  /******
0   123040-123415

TITLE /******
0   123416-123441


END OF ALLOCATION 1103 G039A 09099
```

• DIVIDE CHECK AT 035001

• DIVIDE CHECK AT 035001

SYSTEMS IMPROVED NUMERICAL DIFFERENCING ANALYZER - - - SINDA - - - UNIVAC-1108 FORTRAN-V VERSION PAGE 1

SAMPLE PROBLEM FOR SINDA VERSION 9

****

TIME= 0.00000 DTIMEU= 0.00000 CSGMIN(0)= 0.00000 TEMPCC(0)= 0.00000 RELXCC(0)= 0.00000

T 1=	70.000	T 2=	70.000	T 3=	70.000	T 4=	70.000	T 5=	70.000	T 6=	70.000
T 7=	70.000	T 8=	70.000	T 9=	70.000	T 10=	70.000	T 11=	70.000	T 12=	70.000
T 13=	70.000	T 14=	70.000	T 15=	70.000	T 16=	70.000	T 17=	70.000	T 18=	70.000
T 19=	70.000	T 20=	70.000	T 21=	70.000	T 22=	70.000	T 23=	70.000	T 24=	70.000
T 25=	70.000	T 26=	70.000	T 27=	70.000	T 28=	70.000	T 29=	70.000	T 30=	70.000
T 31=	70.000	T 32=	70.000	T 33=	70.000	T 34=	70.000	T 35=	70.000	T 36=	70.000
T 37=	70.000	T 38=	70.000	T 39=	70.000	T 40=	70.000	T 41=	70.000	T 42=	70.000
T 43=	70.000	T 44=	70.000	T 45=	70.000	T 46=	70.000	T 47=	70.000	T 48=	70.000
T 49=	70.000	T 50=	70.000	T 51=	70.000	T 52=	70.000	T 53=	70.000	T 54=	70.000
T 55=	70.000	T 56=	70.000	T 57=	70.000	T 58=	70.000	T 59=	70.000	T 60=	70.000
T 61=	70.000	T 62=	70.000	T 63=	70.000	T 64=	70.000	T 65=	70.000	T 66=	70.000
T 67=	70.000	T 68=	70.000	T 69=	70.000	T 70=	70.000	T 71=	70.000	T 72=	70.000
T 73=	70.000	T 74=	70.000	T 75=	70.000	T 76=	70.000	T 77=	70.000	T 78=	70.000
T 79=	70.000	T 80=	70.000	T 81=	70.000	T 82=	70.000	T 83=	70.000	T 84=	70.000
T 85=	70.000	T 86=	70.000	T 87=	70.000	T 88=	70.000	T 89=	70.000	T 90=	70.000
T 91=	70.000	T 92=	70.000	T 93=	70.000	T 94=	70.000	T 95=	70.000	T 96=	70.000
T 97=	70.000	T 98=	70.000	T 99=	70.000	T 100=	70.000	T 101=	70.000	T 102=	70.000
T 103=	70.000	T 104=	70.000	T 105=	70.000	T 106=	70.000	T 107=	70.000	T 108=	70.000
T 109=	70.000	T 110=	70.000	T 111=	70.000	T 112=	70.000	T 113=	70.000	T 114=	70.000
T 115=	70.000	T 116=	70.000	T 117=	70.000	T 198=	80.000	T 199=	71.000	T 200=	77.955
T 201=	70.000	T 202=	70.000	T 203=	70.000	T 204=	70.000	T 205=	70.000	T 206=	70.000
T 207=	70.000	T 208=	70.000	T 209=	70.000	T 210=	70.000	T 211=	70.000	T 212=	70.000
T 213=	70.000	T 214=	70.000	T 215=	70.000	T 216=	70.000	T 217=	70.000	T 218=	70.000
T 219=	70.000	T 220=	70.000	T 221=	70.000	T 222=	70.000	T 223=	70.000	T 224=	70.000
T 225=	70.000	T 226=	70.000	T 227=	70.000	T 228=	70.000	T 229=	70.000	T 230=	70.000
T 231=	70.000	T 232=	70.000	T 233=	70.000	T 234=	70.000	T 235=	70.000	T 236=	70.000
T 237=	70.000	T 238=	70.000	T 239=	70.000	T 240=	70.000	T 241=	70.000	T 242=	70.000
T 243=	70.000	T 244=	70.000	T 245=	70.000	T 246=	70.000	T 247=	70.000	T 248=	70.000
T 249=	70.000	T 250=	70.000	T 251=	70.000	T 252=	70.000	T 253=	70.000	T 254=	70.000
T 255=	70.000	T 256=	70.000	T 257=	70.000	T 258=	70.000	T 259=	70.000	T 260=	70.000
T 261=	70.000	T 262=	70.000	T 263=	70.000	T 264=	70.000	T 265=	70.000	T 266=	70.000
T 267=	70.000	T 268=	70.000	T 269=	70.000	T 270=	70.000	T 271=	70.000	T 272=	70.000
T 273=	70.000	T 274=	70.000	T 275=	70.000	T 276=	70.000	T 277=	70.000	T 278=	70.000
T 279=	70.000	T 280=	70.000	T 281=	70.000	T 282=	70.000	T 283=	70.000	T 284=	70.000
T 285=	70.000	T 286=	70.000	T 287=	70.000	T 288=	70.000	T 289=	70.000	T 290=	70.000
T 291=	70.000	T 292=	70.000	T 293=	70.000	T 294=	70.000	T 295=	70.000	T 296=	70.000
T 297=	70.000	T 298=	70.000	T 299=	70.000	T 300=	70.000	T 301=	70.000	T 302=	70.000
T 303=	70.000	T 304=	70.000	T 305=	70.000	T 306=	70.000	T 307=	70.000	T 308=	70.000
T 309=	70.000	T 310=	70.000	T 311=	70.000	T 312=	70.000	T 313=	70.000	T 314=	70.000
T 315=	70.000	T 316=	70.000	T 317=	70.000	T 400=	-459.69				

FLOW RATES (LB/HR)

2448.1	2444.5	2418.5	1211.0	608.01	602.97	608.01	602.97	1211.0	2422.0
26.013	13.024	6.5632	6.4606	6.5632	6.4606	13.024	26.048	2422.0	1211.0
608.01	602.97	608.01	602.97	1211.0	2418.5	26.048	13.024	6.5632	6.4606
6.5632	6.4606	13.024	26.019	2444.5	3.6021	2448.1	2418.5	26.013	

PRESSURE DROPS (LB/FT**2)

251.96	60169.	60556.	82.595	3029.4	3029.4	3029.4	3029.4	82.595	97.225

```
  SAMPLE PROBLEM FOR SINDA VERSION 9
 67757.      .55969-01   1.4937      1.4937      1.4937       1.4937      .55969-01   .44861-01   99.220      82.595
 3029.4      3029.4      3029.4      3029.4      82.595       793.83      .44830-01   .56000-01   1.4936      1.4936
 1.4937      1.4937      .55969-01   .35846      2014.4       .12994+06   201.96      6413.2      3.1847

PRESSURES  (LB/FT**2)
 .13034+06   .13014+06   69977.      9920.5      6411.3       9358.1      6328.7      6229.5      2220.0      2226.6
 2228.1      2226.6      2226.5      6130.3      3100.9       6047.7      3007.2      2216.4      2226.5      2225.0
 2226.4      2216.8      201.96      .00000

VALVE POSITIONS
  .99900      .99000

COMPUTER TIME =      .000 MINUTES

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

* DIVIDE CHECK AT 021506

**********
TIME=  1.00000+00 DTIMEU=  5.00008-03 CSGMIN(    77)=  2.29961-06 TEMPCC(   240)=  2.36740-02 RELXCC(   106)=  5.20706-03
T     1=   71.036    T     2=   52.868    T     3=   37.721    T     4=   24.998    T     5=   14.260    T     6=   14.275
T     7=   71.036    T     8=   52.989    T     9=   37.922    T    10=   25.247    T    11=   14.535    T    12=   14.598
T    13=   71.034    T    14=   52.985    T    15=   37.916    T    16=   25.240    T    17=   14.528    T    18=   14.591
T    19=   71.035    T    20=   52.865    T    21=   37.716    T    22=   24.991    T    23=   14.254    T    24=   14.268
T    25=   14.430    T    26=   5.2795    T    27=  -2.5268    T    28=  -8.7624    T    29=  -14.300    T    30=  -14.261
T    31=   14.426    T    32=   5.3307    T    33=  -2.4408    T    34=  -8.6280    T    35=  -14.122    T    36=  -14.079
T    37=   14.433    T    38=   5.3399    T    39=  -2.4308    T    40=  -8.6162    T    41=  -14.109    T    42=  -14.076
T    43=   14.435    T    44=   5.2848    T    45=  -2.5203    T    46=  -8.7537    T    47=  -14.289    T    48=  -14.251
T    49=   71.033    T    50=   70.676    T    51=   70.321    T    52=   69.966    T    53=   69.613    T    54=   69.615
T    55=   71.033    T    56=   70.677    T    57=   70.322    T    58=   69.969    T    59=   69.617    T    60=   69.619
T    61=   71.034    T    62=   70.678    T    63=   70.323    T    64=   69.970    T    65=   69.618    T    66=   69.619
T    67=   71.034    T    68=   70.677    T    69=   70.321    T    70=   69.967    T    71=   69.614    T    72=   69.616
T    73=   69.621    T    74=   69.269    T    75=   68.919    T    76=   68.569    T    77=   68.210    T    78=   68.212
T    79=   69.621    T    80=   69.270    T    81=   68.920    T    82=   68.572    T    83=   68.224    T    84=   68.226
T    85=   69.620    T    86=   69.269    T    87=   68.919    T    88=   68.571    T    89=   68.224    T    90=   68.224
T    91=   69.621    T    92=   69.269    T    93=   68.918    T    94=   68.568    T    95=   68.220    T    96=   68.222
T    97=   71.029    T    98=   71.029    T    99=   71.031    T   100=   71.032    T   101=   14.411    T   102=   14.413
T   103=   14.414    T   104=   14.420    T   105=  -14.160    T   106=  -14.137    T   107=   71.030    T   108=   71.031
T   109=   69.617    T   110=   69.618    T   111=   69.618    T   112=   69.619    T   113=   68.224    T   114=   68.223
T   115=   40.059    T   116=   71.051    T   117=   40.107    T   198=   85.000    T   199=   44.039    T   200=   71.033
T   201=   71.037    T   202=   41.680    T   203=   27.659    T   204=   15.804    T   205=   5.6906    T   206=   14.277
T   207=   71.036    T   208=   41.990    T   209=   28.039    T   210=   16.233    T   211=   6.1963    T   212=   14.552
T   213=   71.035    T   214=   41.922    T   215=   28.029    T   216=   16.223    T   217=   6.1354    T   218=   14.544
T   219=   71.036    T   220=   41.672    T   221=   27.649    T   222=   15.794    T   223=   5.6793    T   224=   14.270
T   225=   14.432    T   226=  -2.7205    T   227=  -10.099    T   228=  -18.842    T   229=  -23.125    T   230=  -14.264
T   231=   14.429    T   232=  -2.4550    T   233=  -9.7796    T   234=  -18.633    T   235=  -22.965    T   236=  -14.276
T   237=   14.437    T   238=  -2.4391    T   239=  -9.7614    T   240=  -18.673    T   241=  -22.950    T   242=  -14.263
T   243=   14.437    T   244=  -2.7085    T   245=  -10.074    T   246=  -18.826    T   247=  -23.123    T   248=  -14.265
```

	249=	71.034	T	250=	67.486	T	251=	67.142	T	252=	66.799	T	253=	66.457	T	254=	69.615
T	255=	71.033	T	256=	67.495	T	257=	67.152	T	258=	66.810	T	259=	66.469	T	260=	69.619
T	261=	71.035	T	262=	67.496	T	263=	67.153	T	264=	66.811	T	265=	66.470	T	266=	69.623
T	267=	71.035	T	268=	67.487	T	269=	67.143	T	270=	66.800	T	271=	66.458	T	272=	69.616
T	273=	69.622	T	274=	66.125	T	275=	65.786	T	276=	65.448	T	277=	67.855	T	278=	68.213
T	279=	69.621	T	280=	66.133	T	281=	65.795	T	282=	65.458	T	283=	65.122	T	284=	68.226
T	285=	69.620	T	286=	66.133	T	287=	65.794	T	288=	65.457	T	289=	65.121	T	290=	68.225

```
     SAMPLE PROBLEM FOR SINDA VERSION 9

T  291=  69.621   T  292=  66.124   T  293=  65.785   T  294=  65.447   T  295=  65.110   T  296=  68.222
T  297=  71.029   T  298=  71.029   T  299=  71.031   T  300=  71.033   T  301=  14.412   T  302=  14.413
T  303=  14.415   T  304=  14.421   T  305= -14.157   T  306= -14.136   T  307=  71.030   T  308=  71.032
T  309=  69.618   T  310=  69.618   T  311=  69.618   T  312=  69.619   T  313=  68.224   T  314=  68.223
T  315=  40.059   T  316=  71.052   T  317=  40.108   T  400= -459.69

FLOW RATES (LB/HR)
   2465.7    2962.2    874.83    437.32    219.83    217.50    219.77    217.44    437.21    874.53
   1587.3    793.81    397.56    396.25    397.56    396.25    793.81    1587.6    874.53    437.36
   219.99    217.38    219.88    217.29    437.18    874.83    1587.6    793.81    397.56    396.25
   397.56    396.25    793.81    1587.3    2962.2    3.5934    2465.8    874.83    1587.3

PRESSURE DROPS (LB/FT**2)
   204.28    60998.    65467.    13.867    379.26    379.26    379.06    379.06    14.069    16.904
   58619.    39.344    3626.1    3626.1    3626.1    3626.1    39.349    47.268    16.904    14.077
   370.63    370.63    370.33    370.33    14.375    138.20    47.269    39.348    3622.0    3622.0
   3622.0    3622.0    39.353    378.18    2045.2  .12941+06   205.05    811.92    7420.0

PRESSURES (LB/FT**2)
  .12982+06 .12962+06  68668.    3200.4    2821.1    3186.2    2807.0    2790.1    10048.    6423.7
   10011.    6384.4    6337.1    2773.2    2402.8    2759.1    2388.5    2250.3    6289.8    2667.8
   6250.5    2628.5    205.05    .00000

VALVE POSITIONS
   .99900    .34220

COMPUTER TIME =    1.952 MINUTES

**********
TIME=  2.00000+00 DTIMEU=  5.00008-03 CSGMIN(   77)=  2.29959-06 TEMPCC(   11)=  1.10626-03 RELXCC(    9)=  1.33514-04
T    1=  70.978   T    2=  52.806   T    3=  37.648   T    4=  24.902   T    5=  14.130   T    6=  14.130
T    7=  70.978   T    8=  52.929   T    9=  37.852   T   10=  25.156   T   11=  14.411   T   12=  14.411
T   13=  70.978   T   14=  52.926   T   15=  37.847   T   16=  25.150   T   17=  14.404   T   18=  14.404
T   19=  70.978   T   20=  52.803   T   21=  37.643   T   22=  24.896   T   23=  14.123   T   24=  14.123
T   25=  14.268   T   26=  5.0660   T   27= -2.8035   T   28= -9.1347   T   29= -14.745   T   30= -14.745
T   31=  14.268   T   32=  5.1233   T   33= -2.7106   T   34= -8.9943   T   35= -14.566   T   36= -14.566
T   37=  14.268   T   38=  5.1254   T   39= -2.7072   T   40= -8.9889   T   41= -14.560   T   42= -14.560
T   43=  14.268   T   44=  5.0681   T   45= -2.8002   T   46= -9.1244   T   47= -14.739   T   48= -14.739
T   49=  70.978   T   50=  70.621   T   51=  70.265   T   52=  69.911   T   53=  69.558   T   54=  69.558
T   55=  70.978   T   56=  70.622   T   57=  70.268   T   58=  69.914   T   59=  69.562   T   60=  69.562
T   61=  70.978   T   62=  70.622   T   63=  70.268   T   64=  69.914   T   65=  69.562   T   66=  69.562
T   67=  70.978   T   68=  70.621   T   69=  70.265   T   70=  69.911   T   71=  69.558   T   72=  69.558
T   73=  69.560   T   74=  69.208   T   75=  68.858   T   76=  68.508   T   77=  68.149   T   78=  68.149
T   79=  69.560   T   80=  69.209   T   81=  68.860   T   82=  68.512   T   83=  68.165   T   84=  68.165
T   85=  69.560   T   86=  69.209   T   87=  68.860   T   88=  68.512   T   89=  68.165   T   90=  68.165
T   91=  69.560   T   92=  69.208   T   93=  68.858   T   94=  68.508   T   95=  68.149   T   96=  68.165
T   97=  70.978   T   98=  70.978   T   99=  70.978   T  100=  70.978   T  101=  14.264   T  102=  14.268
T  103=  14.268   T  104=  14.268   T  105= -14.655   T  106= -14.652   T  107=  70.978   T  108=  70.978
T  109=  69.560   T  110=  69.560   T  111=  69.560   T  112=  69.560   T  113=  68.162   T  114=  68.160
T  115=  39.834   T  116=  70.978   T  117=  39.920   T  198=  80.000   T  199=  43.292   T  200=  70.978
T  201=  70.978   T  202=  41.611   T  203=  27.574   T  204=  15.685   T  205=  5.5215   T  206=  14.130
```

SAMPLE PROBLEM FOR SINDA VERSION 9

```
T  207=  70.978   T  208=  41.922   T  209=  27.956   T  210=  16.117   T  211=  5.9828   T  212=  14.410
T  213=  70.978   T  214=  41.915   T  215=  27.946   T  216=  16.107   T  217=  5.9716   T  218=  14.404
T  219=  70.978   T  220=  41.604   T  221=  27.565   T  222=  15.674   T  223=  5.5096   T  224=  14.123
T  225=  14.268   T  226= -2.9976   T  227= -10.468   T  228= -19.350   T  229= -23.681   T  230= -14.745
T  231=  14.268   T  232= -2.7225   T  233= -10.147   T  234= -19.188   T  235= -23.511   T  236= -14.566
T  237=  14.268   T  238= -2.7129   T  239= -10.135   T  240= -19.183   T  241= -23.506   T  242= -14.560
T  243=  14.267   T  244= -2.9876   T  245= -10.457   T  246= -19.346   T  247= -23.676   T  248= -14.739
T  249=  70.978   T  250=  67.432   T  251=  67.088   T  252=  66.746   T  253=  66.404   T  254=  69.558
T  255=  70.978   T  256=  67.442   T  257=  67.099   T  258=  66.757   T  259=  66.416   T  260=  69.562
T  261=  70.978   T  262=  67.442   T  263=  67.099   T  264=  66.757   T  265=  66.416   T  266=  69.562
T  267=  70.978   T  268=  67.432   T  269=  67.088   T  270=  66.746   T  271=  66.404   T  272=  69.558
T  273=  69.560   T  274=  66.066   T  275=  65.727   T  276=  65.389   T  277=  67.794   T  278=  68.149
T  279=  69.560   T  280=  66.075   T  281=  65.737   T  282=  65.400   T  283=  65.069   T  284=  68.165
T  285=  69.560   T  286=  66.075   T  287=  65.737   T  288=  65.400   T  289=  65.069   T  290=  68.165
T  291=  69.560   T  292=  66.066   T  293=  65.727   T  294=  65.389   T  295=  65.052   T  296=  68.160
T  297=  70.978   T  298=  70.978   T  299=  70.978   T  300=  70.978   T  301=  14.269   T  302=  14.268
T  303=  14.268   T  304=  14.268   T  305= -14.655   T  306= -14.652   T  307=  70.978   T  308=  70.978
T  309=  69.560   T  310=  69.560   T  311=  69.560   T  312=  69.560   T  313=  68.162   T  314=  68.160
T  315=  39.834   T  316=  70.978   T  317=  39.880   T  400= -459.69

FLOW RATES (LB/HR)
  2465.7     2462.2     874.68     437.41     219.91     217.50     219.85     217.44     437.29     874.70
  1587.5     793.74     397.52     396.22     397.52     396.22     793.74     1587.5     874.70     437.44
  220.02     217.42     219.93     217.33     437.26     874.68     1587.5     793.74     397.52     396.21
  397.52     396.21     793.74     1587.5     2462.2     3.6015     2465.8     874.68     1587.5

PRESSURE DROPS  (LB/FT**2)
  204.28     60948.     65468.     13.867     379.23     379.23     379.03     379.03     14.570     16.906
  58619.     39.344     3626.0     3626.0     3626.0     3626.0     39.348     47.268     16.906     14.079
  370.61     370.61     370.30     370.30     14.382     338.27     47.269     39.348     3621.8     3621.8
  3621.8     3621.8     39.352     378.18     2045.3     .12941+06  205.05     811.58     7421.1

PRESSURES  (LB/FT**2)
  .12982+06  .12962+06  68668.     3200.2     2821.2     3186.4     2807.2     2790.2     10050.     6423.7
  10010.     6384.3     6337.0     2773.3     2403.0     2759.3     2388.7     2250.4     6289.8     2667.9
  6250.4     2628.6     205.05     .00000

VALVE POSITIONS
  .99900     .34220

COMPUTER TIME =    2.922 MINUTES

**********
TIME=  3.00000+00 DTIMEU=  5.00000-03 CSGMIN(   77)=  2.29959-06 TEMPCC(   11)=  4.99725-04 RELXCC(   21)=  6.10352-05
T    1=  70.978   T    2=  52.804   T    3=  37.648   T    4=  24.903   T    5=  14.135   T    6=  14.135
T    7=  70.978   T    8=  52.929   T    9=  37.851   T   10=  25.155   T   11=  14.415   T   12=  14.415
T   13=  70.978   T   14=  52.926   T   15=  37.846   T   16=  25.149   T   17=  14.403   T   18=  14.403
T   19=  70.978   T   20=  52.803   T   21=  37.643   T   22=  24.896   T   23=  14.123   T   24=  14.123
T   25=  14.267   T   26=  5.0658   T   27= -2.8039   T   28= -9.1350   T   29= -14.746   T   30= -14.744
T   31=  14.267   T   32=  5.1231   T   33= -2.7110   T   34= -8.9945   T   35= -14.567   T   36= -14.567
T   37=  14.267   T   38=  5.1251   T   39= -2.7076   T   40= -8.9932   T   41= -14.560   T   42= -14.560
```

SAMPLE PROBLEM FOR SINDA VERSION 9

T	43=	14.267	T	44=	5.0678	T	45=	-2.8006	T	46=	-4.1297	T	47=	-14.739	T	48=	-14.739
T	49=	70.978	T	50=	70.621	T	51=	70.265	T	52=	69.911	T	53=	69.558	T	54=	69.558
T	55=	70.978	T	56=	70.622	T	57=	70.268	T	58=	69.914	T	59=	69.562	T	60=	69.562
T	61=	70.978	T	62=	70.622	T	63=	70.267	T	64=	69.914	T	65=	69.562	T	66=	69.562
T	67=	70.978	T	68=	70.621	T	69=	70.265	T	70=	69.911	T	71=	69.558	T	72=	69.558
T	73=	69.560	T	74=	69.208	T	75=	68.857	T	76=	68.508	T	77=	68.149	T	78=	68.149
T	79=	69.560	T	80=	69.209	T	81=	68.860	T	82=	68.511	T	83=	68.164	T	84=	68.164
T	85=	69.560	T	86=	69.209	T	87=	68.860	T	88=	68.511	T	89=	68.164	T	90=	68.164
T	91=	69.560	T	92=	69.208	T	93=	68.858	T	94=	68.508	T	95=	68.160	T	96=	68.160
T	97=	70.978	T	98=	70.978	T	99=	70.978	T	100=	70.978	T	101=	14.264	T	102=	14.267
T	103=	14.267	T	104=	14.267	T	105=	-14.656	T	106=	-14.652	T	107=	70.978	T	108=	70.978
T	109=	69.560	T	110=	69.560	T	111=	69.560	T	112=	69.560	T	113=	68.162	T	114=	68.159
T	115=	39.833	T	116=	70.978	T	117=	39.880	T	198=	80.000	T	199=	43.892	T	200=	70.978
T	201=	70.978	T	202=	41.611	T	203=	27.574	T	204=	15.685	T	205=	5.5207	T	206=	14.135
T	207=	70.978	T	208=	41.922	T	209=	27.956	T	210=	16.117	T	211=	5.9823	T	212=	14.410
T	213=	70.978	T	214=	41.915	T	215=	27.946	T	216=	16.107	T	217=	5.9716	T	218=	14.403
T	219=	70.978	T	220=	41.603	T	221=	27.565	T	222=	15.674	T	223=	5.5093	T	224=	14.123
T	225=	14.267	T	226=	-2.9978	T	227=	-10.469	T	228=	-19.350	T	229=	-23.681	T	230=	-14.746
T	231=	14.267	T	232=	-2.7227	T	233=	-10.147	T	234=	-19.188	T	235=	-23.512	T	236=	-14.567
T	237=	14.267	T	238=	-2.7131	T	239=	-10.136	T	240=	-19.184	T	241=	-23.506	T	242=	-14.560
T	243=	14.267	T	244=	-2.9878	T	245=	-10.457	T	246=	-19.346	T	247=	-23.676	T	248=	-14.739
T	249=	70.978	T	250=	67.432	T	251=	67.088	T	252=	66.745	T	253=	66.404	T	254=	69.558
T	255=	70.978	T	256=	67.441	T	257=	67.098	T	258=	66.757	T	259=	66.416	T	260=	69.562
T	261=	70.978	T	262=	67.441	T	263=	67.098	T	264=	66.757	T	265=	66.416	T	266=	69.562
T	267=	70.978	T	268=	67.432	T	269=	67.088	T	270=	66.745	T	271=	66.404	T	272=	69.558
T	273=	69.560	T	274=	66.066	T	275=	65.726	T	276=	65.389	T	277=	67.739	T	278=	68.149
T	279=	69.560	T	280=	66.075	T	281=	65.737	T	282=	65.400	T	283=	65.064	T	284=	68.164
T	285=	69.560	T	286=	66.075	T	287=	65.737	T	288=	65.400	T	289=	65.064	T	290=	68.164
T	291=	69.560	T	292=	66.066	T	293=	65.726	T	294=	65.389	T	295=	65.052	T	296=	68.160
T	297=	70.978	T	298=	70.978	T	299=	70.978	T	300=	70.978	T	301=	14.264	T	302=	14.267
T	303=	14.267	T	304=	14.267	T	305=	-14.656	T	306=	-14.652	T	307=	70.978	T	308=	70.978
T	309=	69.560	T	310=	69.560	T	311=	69.560	T	312=	69.560	T	313=	68.162	T	314=	68.159
T	315=	39.833	T	316=	70.978	T	317=	39.880	T	400=	-459.69						

FLOW RATES (LB/HR)

2465.7	2462.2	874.70	437.40	219.88	217.52	219.82	217.46	437.29	874.68
1587.5	793.73	397.52	396.21	397.52	396.21	793.73	1587.5	874.68	437.43
220.02	217.41	219.93	217.32	437.25	874.70	1587.5	793.73	397.52	396.21
397.52	396.21	793.73	1587.5	2462.2	3.5933	2465.8	874.70	1587.5	

PRESSURE DROPS (LB/FT**2)

204.28	60949.	65468.	13.867	379.23	379.23	379.02	379.02	14.070	16.956
59619.	39.344	3626.0	3626.0	3626.0	3626.0	39.349	47.268	14.906	14.078
370.60	370.60	370.30	370.30	14.382	138.27	47.258	39.348	3621.9	3621.8
3621.8	3621.8	39.352	378.18	2045.3	.12941+06	205.05	811.65	7421.1	

PRESSURES (LB/FT**2)

.12982+06	.12962+06	68668.	3200.3	2821.2	3186.4	2807.1	2790.2	10050.	6423.6
10010.	6324.3	6337.0	2773.3	2403.0	2759.2	2388.7	2250.4	6239.7	2667.9
6250.4	2628.6	205.05	.00000						

VALVE POSITIONS

.99900 .34220

SAMPLE PROBLEM FOR SINDA VERSION 9

COMPUTER TIME = 3.886 MINUTES

END OF DATA

● XOT CUR   25 FEB 73   14:20:22

14:20:22

1. TRW T

END CUR LCC 1102-0039 L9

FIGURE 12

RADIATOR TEMPERATURES FOR SAMPLE PROBLEM

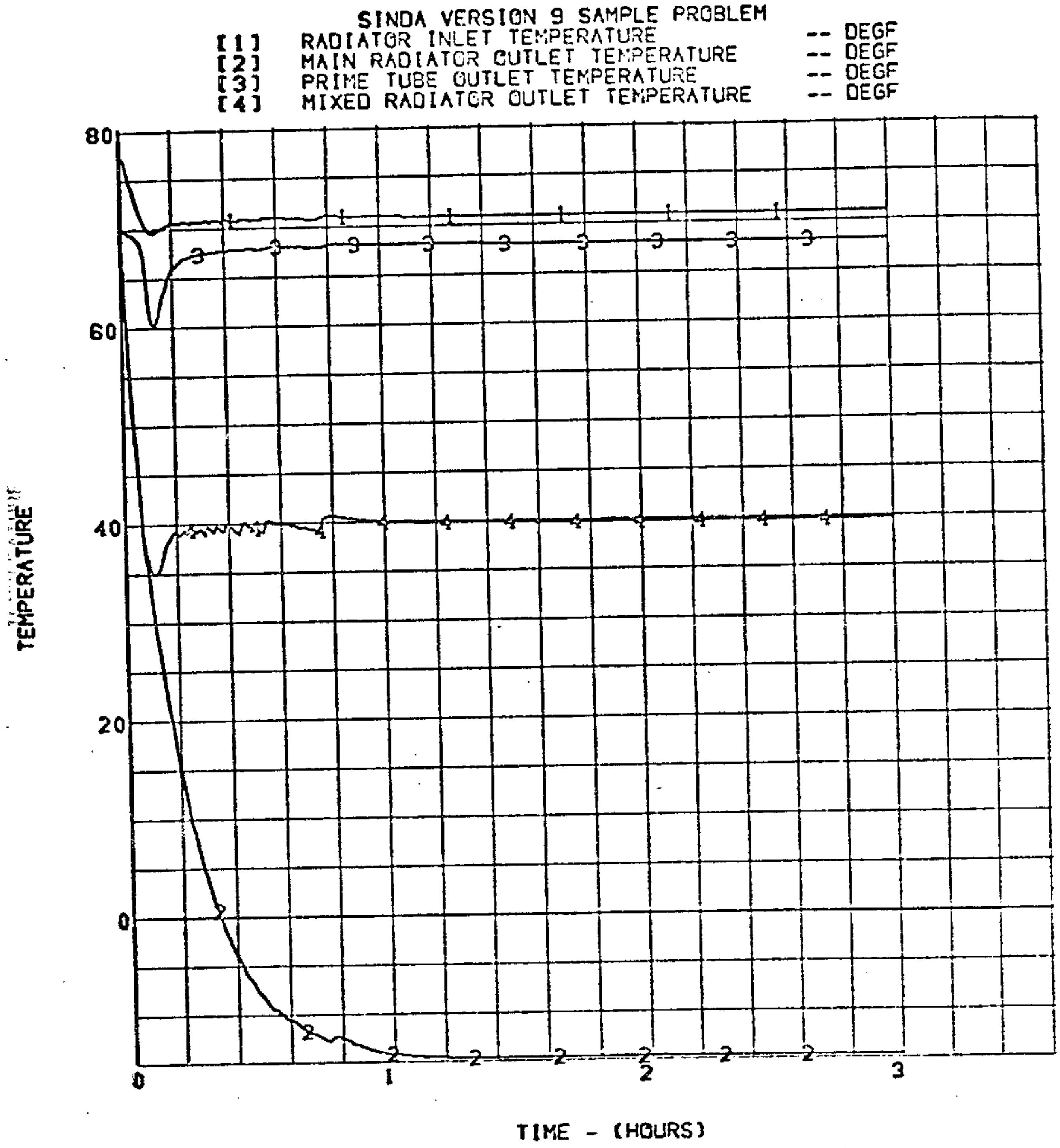


FIGURE 13

SYSTEM TEMPERATURES FOR SAMPLE PROBLEM

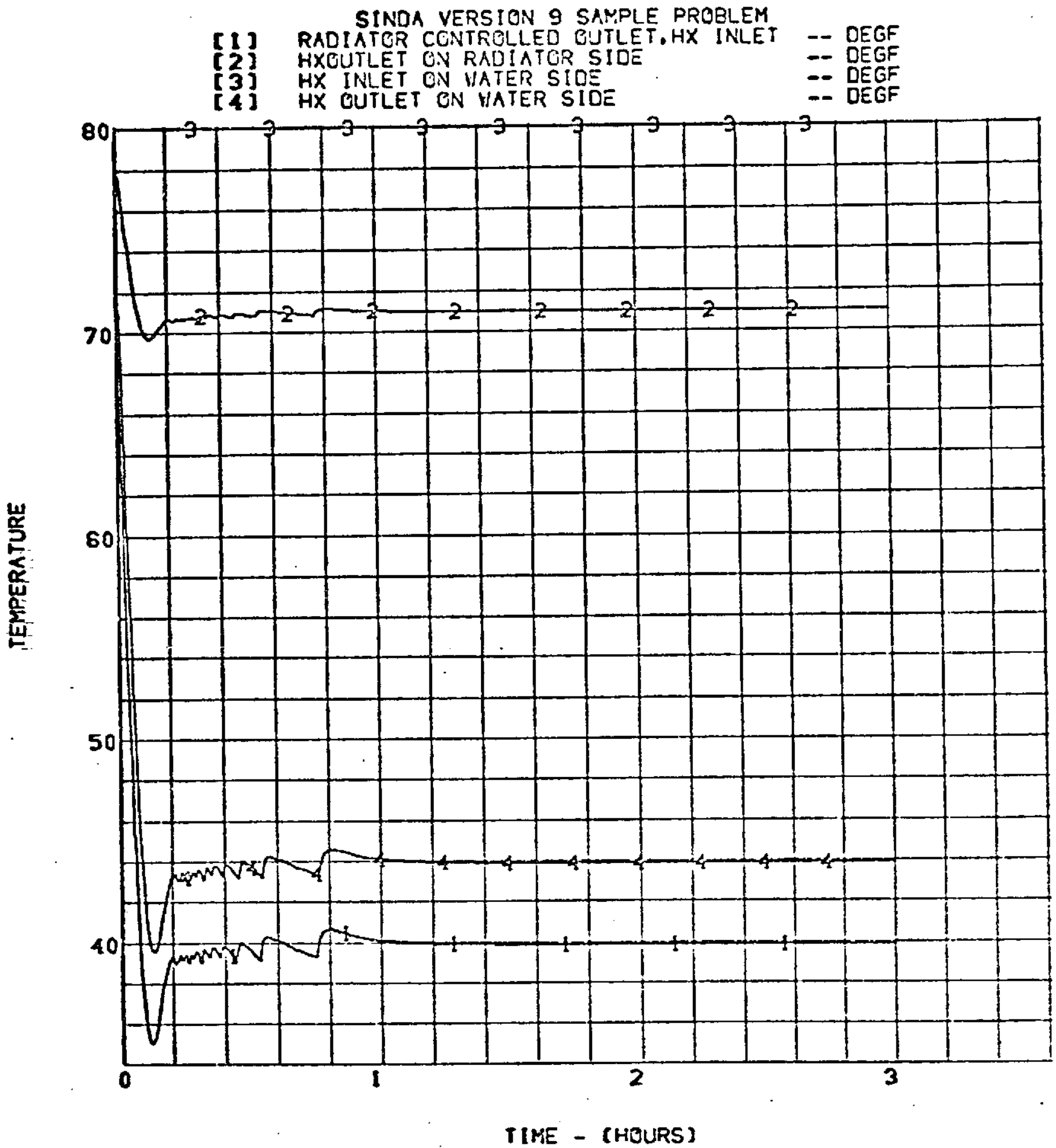


FIGURE 14

SYSTEM PRESSURES FOR SAMPLE PROBLEM

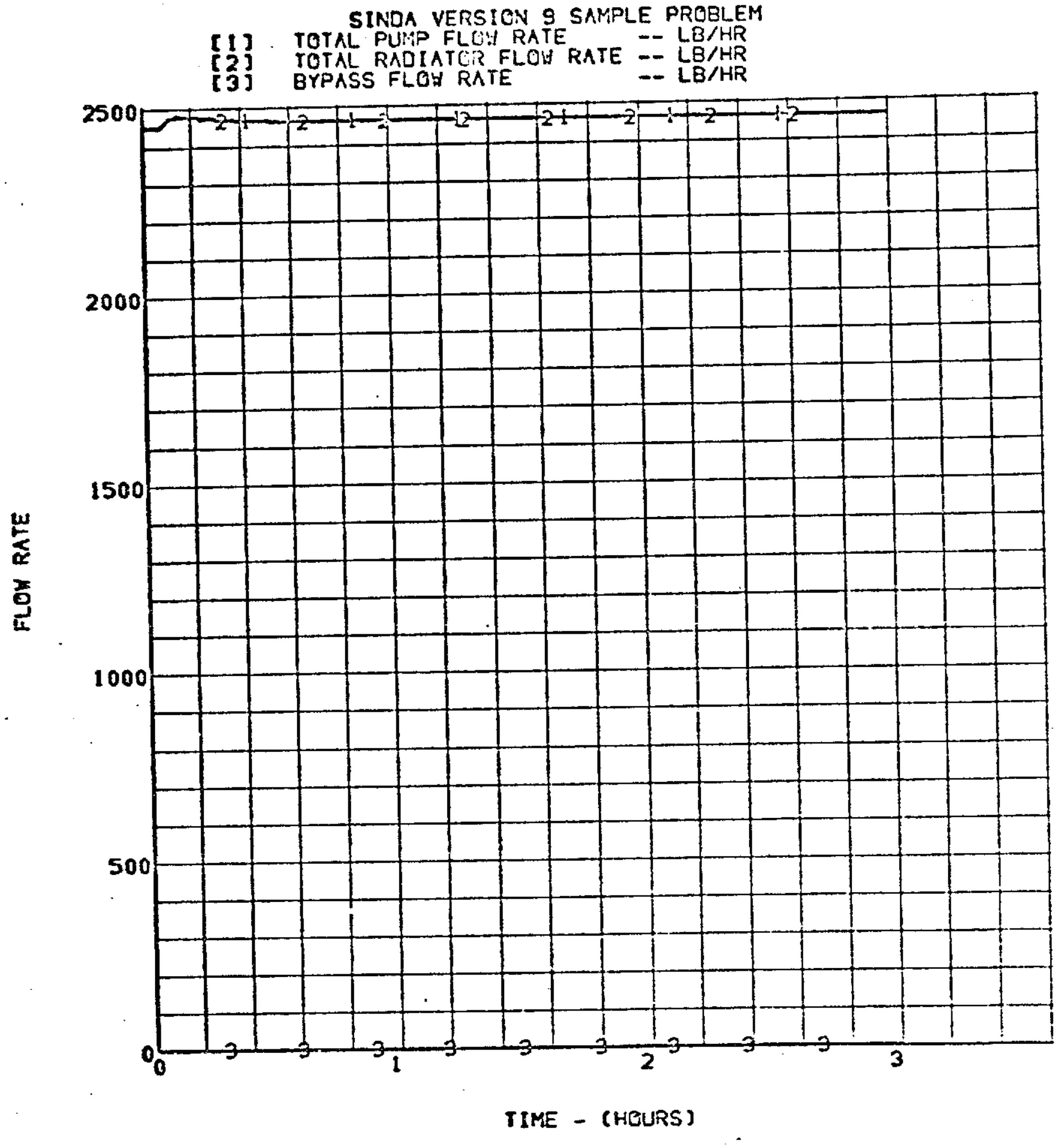


FIGURE 15

RADIATOR FLOW RATES FOR SAMPLE PROBLEM

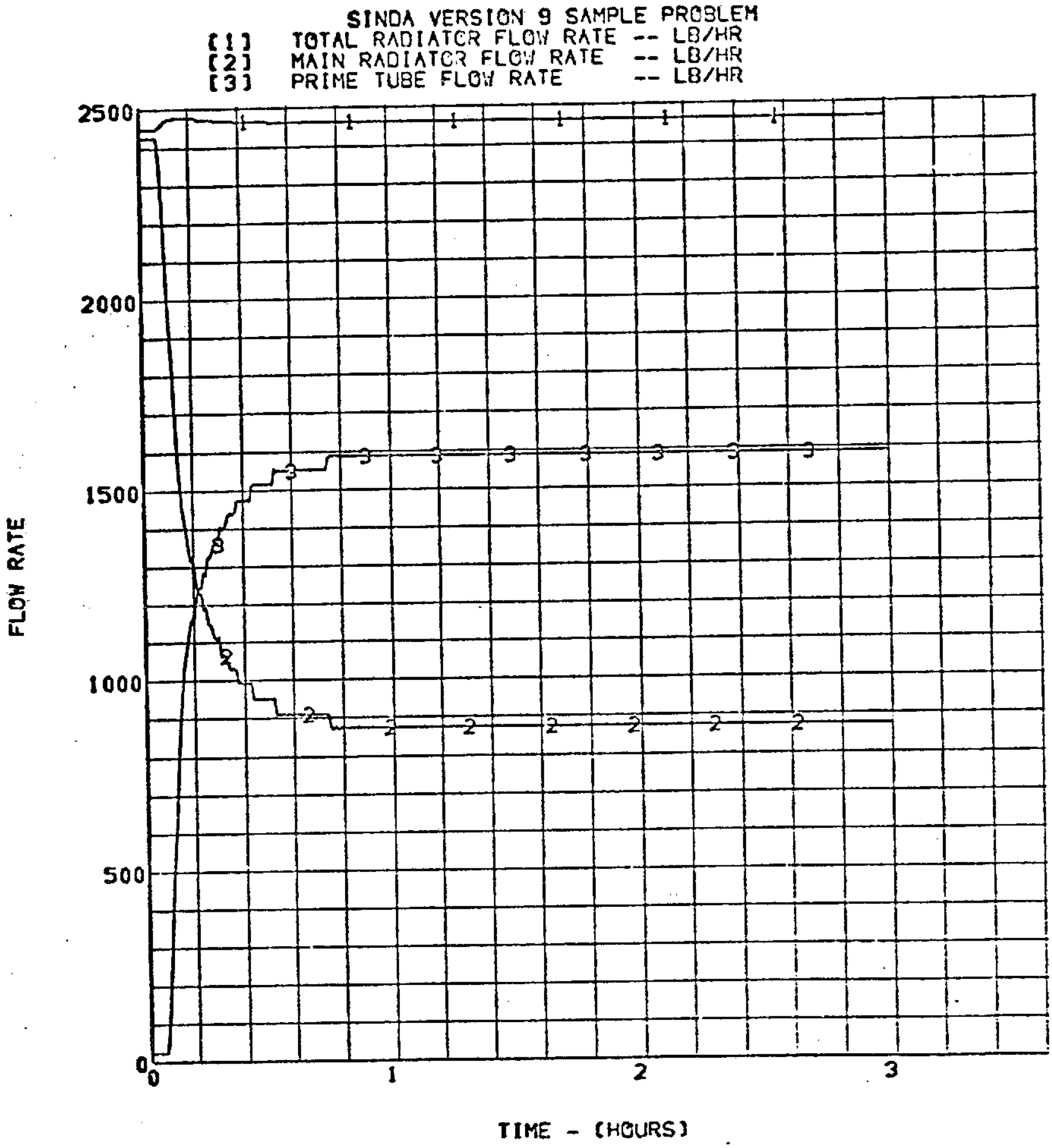


FIGURE 16

PUMP PRESSURES FOR SAMPLE PROBLEM

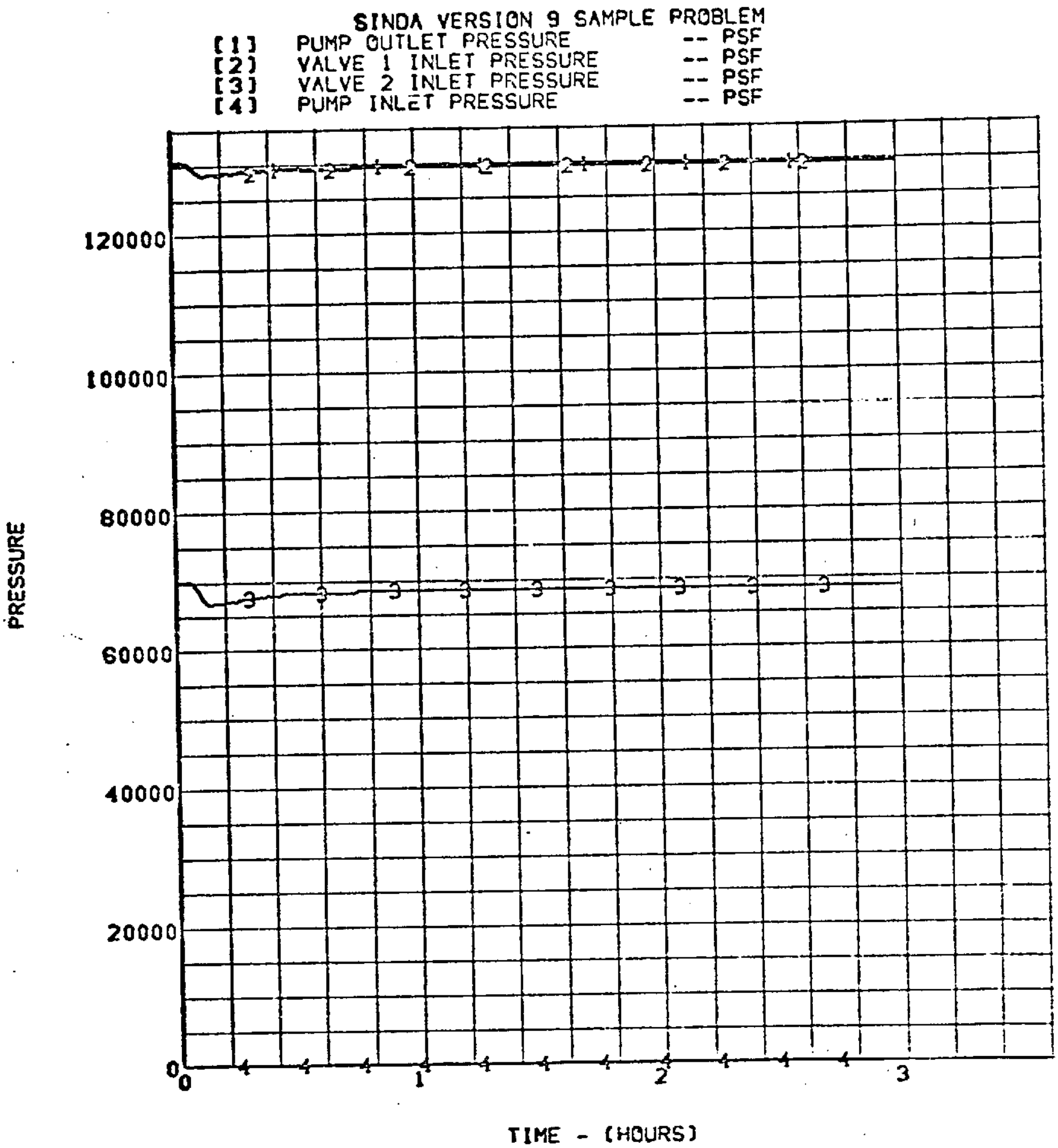
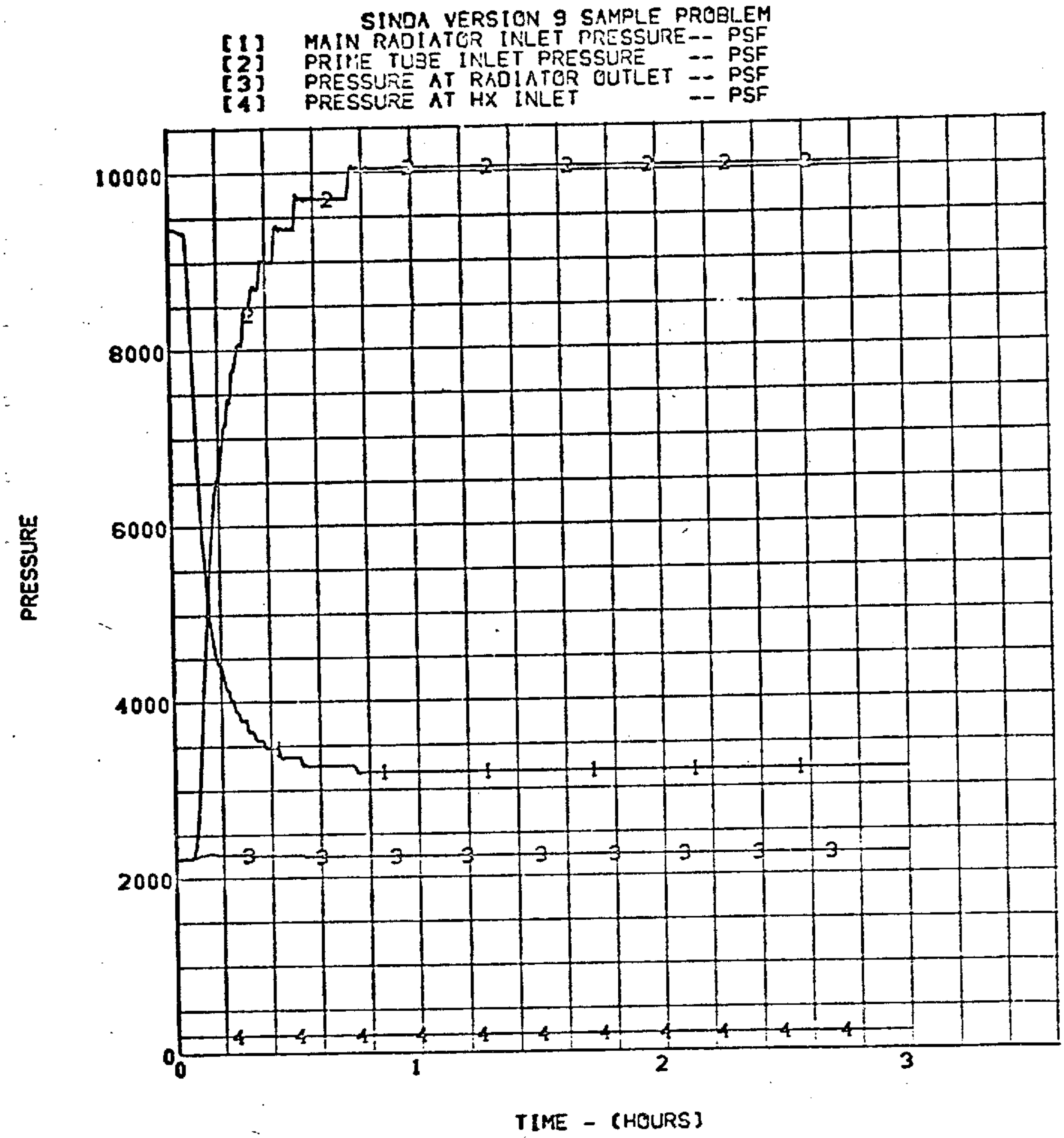


FIGURE 17

SYSTEM PRESSURES FOR SAMPLE PROBLEM



6.0 REFERENCES

1. Oren, J. A., Phillips, M. A., and Williams, D. R.; "Modular Thermal Analysis Routine", LTV Report 00.1524, Vol. I, 27 March 1972.
2. Sellers, J. R., Tribus, M., and Klein, S. J.; "Heat Transfer to Laminar Flow in a Round Tube or Flat Conduit - The Graetz Problem Extended", Transaction of ASME, Vol. 78, 1956, pp 441-448.
3. Echert, E.R.G. and Drake, R. M.; <u>Heat and Mass Transfer</u>, McGraw Hill, New York, 1959.
4. Hardi, P. D., Howell, H. R., Williams, J. H., "Lunar Module Ascent Stage Thermal Simular", LTV Report No. 350.3, 11 August 1957
5. Sparrow, E. M., and Cess, R.D., <u>Radiation Heat Transfer</u>, Brooks/Cole Publishing Co., Belmont, California, 1966.
6. Gaddis, J. L., "Explicit Finite Difference Heat Transfer Program - LVVM25", LTV Report No. 00.823, 29 July 1966.
7. Smith, J. P., "SINDA Users Manual", TRW Report 14690-H001-R0-00, April 1971.

APPENDIX A

SUBROUTINE LISTINGS

A Fortran listing is presented below for the subroutines which were modified or added to the SINDA preprocessor and user subroutine library which create SINDA/VERSION 9.

```
      SUBROUTINE CODERD                                                 CDR   1    CODERD
C                                                                       CDR   2    CODERD
C        SUBROUTINE CODERD (CODE READ) READS THE TITLE BLOCK AND THE    CDR   3    CODERD
C        NEXT FOUR BLOCK HEADER CARDS.  IE. BCD 3NODE DATA,             CDR   4    CODERD
C        BCD 3CONDUCTOR DATA, BCD 3CONSTANTS DATA, AND                  CDR   5    CODERD
C        BCD 3ARRAY DATA.                                               CDR   6    CODERD
C                                                                       CDR   7    CODERD
      COMMON/JPS/JPSTOT                                                 VERS 7     CODERD
      COMMON /BUCKET/ B(1)                                              CDR   8    CODERD
      COMMON /LOGIC/ LNODE,LCOND,LCONST,LARRAY,LPRINT,KBRNCH            CDR   9    CODERD
     1   ,IFIXC(50),KTPRNT,AYPRNT,GENERL,LQ,LONG2                       VERS-005   CODERD
      COMMON /TAPE/  NIN, NOUT, INTERN, LB3D, LB4P, LUT1,LUT2,LUT3,LUT4 CDR  11    CODERD
      COMMON /DATA/ NNO,NNA,NNB,NNT,NGL,NGR,NGT,NUC,NEC1,NEC2,NCT,LENA, CDR  12    CODERD
     1     ERDATA,PROGRM,ENDRUN,LSEQ1,LSEQ2,LONG                        CDR  13    CODERD
      COMMON /PLOGIC/  PARINT, PARFIN, PNODE, PCOND, PCONST, PARRAY,    CDR  14    CODERD
     1                 PTITLE, PCHGID                                   CDR  15    CODERD
      COMMON /POINT/ LOC(20),LEN(20),LENBKT,TITLE(20)                   CDR  16    CODERD
      INTEGER  ALPH, THERM, PCSLNG, CODE, END, REMARK, TEMPB, CONDB,    CDR  17    CODERD
     1       CONSTB, ARRYB, ENDDAT, PRINT, HINIT, GENRLP, BLANK, TITLE,CDR  18    CODERD
     2       FINE, PCHGID                                               CDR  19    CODERD
      INTEGER  COL1,COMMNT,PCSSHT,BLOCK                                 CDR  20    CODERD
      INTEGER QB,ENDPRM                                                 CDR  21    CODERD
      LOGICAL LNODE,LLOGIC,LEND,NOREAD                                  CDR  22    CODERD
      LOGICAL  LCOND, LCONST, LARRAY, LPRINT, GENERL, LONG              CDR  23    CODERD
      LOGICAL   PARINT, PARFIN, PNODE, PCOND, PCONST, PARRAY, PTITLE    CDR  24    CODERD
      LOGICAL KTPRNT,AYPRNT,LQ,LONG2                                    VERS-005   CODERD
      DIMENSION ALPH(14), IB(1), BLOCK(4), LLOGIC(1)                    CDR  26    CODERD
      DIMENSION FIXC(1)                                                 VERS-005   CODERD
      EQUIVALENCE (IFIXC,FIXC)                                          VERS-005   CODERD
      EQUIVALENCE (B,IB), (LLOGIC,LNODE)                                CDR  27    CODERD
      DATA  END /6HEND   /,BLANK/6H      /                              CDR  28    CODERD
      DATA  THERM/6HTHERMA/ ,  PCSLNG/6HL LPCS/                         CDR  29    CODERD
      DATA  REMARK/6HREM   /, PRINT/6H *    /                           CDR  30    CODERD
      DATA  TEMPB /6HNODE D/,CONDB/6HCONDUC/, CONSTB/6HCONSTA/,ARRYB    CDR  31    CODERD
     1      /6HARRAY /, GENRLP/6HGENERA/, HINIT/6HINITIA/, ENDDAT       CDR  32    CODERD
     2      /6HEND OF/, FINE/6HFINAL /                                  CDR  33    CODERD
      DATA COMMNT /1HC/, PCSSHT /6HL SPCS/, ITWO /1H2/                  CDR  34    CODERD
      DATA (BLOCK(I),I=1,4) /6HNODE D, 6HCONDUC, 6HCONSTA, 6HARRAY /    CDR  35    CODERD
      DATA QB/6HSOURCE/                                                 CDR  36    CODERD
      DATA ENDPRM/6HEND PA/                                             CDR  37    CODERD
      COMMON/SRDCOM/COL1,COL27,ALPH,CODE,N                                    V 6  CODERD
C                                                                       CDR  38    CODERD
C        INITIALIZATION                                                 CDR  39    CODERD
C                                                                       CDR  40    CODERD
      ILAST=0                                                           CDR  41    CODERD
      PTITLE=.FALSE.                                                    CDR  42    CODERD
      PNODE=.FALSE.                                                     CDR  43    CODERD
      PCOND=.FALSE.                                                     CDR  44    CODERD
      PCONST=.FALSE.                                                    CDR  45    CODERD
      PARRAY=.FALSE.                                                    CDR  46    CODERD
      LPRINT=.FALSE.                                                    CDR  47    CODERD
      KTPRNT=.FALSE.                                                    CDR  48    CODERD
      AYPRNT=.FALSE.                                                    CDR  49    CODERD
      PARINT=.FALSE.                                                    CDR  50    CODERD
      PARFIN=.FALSE.                                                    CDR  51    CODERD
      LONG2=.FALSE.                                                     VERS-005   CODERD
      LQ=.FALSE.                                                        CDR  52    CODERD
C                                                                       CDR  53    CODERD
C        READ BCD 3THERMAL/GENERAL CARD                                 CDR  54    CODERD
```

```
      C                                                                   COR  55     CODERD
   10 CONTINUE                                                            COR  56     CODERD
      CALL SREADC(1)                                                          V 6     CODERD
      IF (COL1.NE.COMMNT) GO TO 20                                        COR  58     CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                              COR  59     CODERD
      GO TO 10                                                            COR  60     CODERD
   20 CONTINUE                                                            COR  61     CODERD
      IF (ALPH(3).EQ.ENDDAT) GO TO 520                                    COR  62     CODERD
      IF (ALPH(3).EQ.ENDPRM) GO TO 10                                     COR  63     CODERD
      WRITE (NOUT,620)                                                    COR  64     CODERD
      WRITE (NOUT,660) ALPH                                               COR  65     CODERD
C                                                                         COR  66     CODERD
C        DEBUG PRINT IF * IN COLUMN 80                                    COR  67     CODERD
C                                                                         COR  68     CODERD
      IF (ALPH(14).EQ.PRINT) LPRINT=.TRUE.                                COR  69     CODERD
      IF (ALPH(3).NE.THERM) GO TO 60                                      COR  70     CODERD
C                                                                         COR  71     CODERD
C        THERMAL PROBLEM - CHECK FOR LONG OR SHORT PSEUDO COMPUTE SEQ.    COR  72     CODERD
C                                                                         COR  73     CODERD
      IF (ALPH(4).NE.PCSLNG) GO TO 30                                     COR  74     CODERD
      LONG=.TRUE.                                                         COR  75     CODERD
      IF (ALPH(5).EQ.ITWO) LONG2=.TRUE.                                   VERS-005    CODERD
      GO TO 60                                                            COR  76     CODERD
   30 IF (ALPH(4).NE.PCSSHT) GO TO 500                                    COR  77     CODERD
      GO TO 80                                                            COR  78     CODERD
C                                                                         COR  79     CODERD
C        CHECK FOR INITIAL PARAMETER RUN                                  COR  80     CODERD
C                                                                         COR  81     CODERD
   40 CONTINUE                                                            COR  82     CODERD
      IF (ALPH(3).NE.HINIT) GO TO 50                                      COR  83     CODERD
      PARINT=.TRUE.                                                       COR  84     CODERD
      PCHGID=HINIT                                                        COR  85     CODERD
      WRITE (LB3D) (HINIT,I=1,50)                                         VERS   3    CODERD
      CALL INCORE (0)                                                     COR  87     CODERD
      GO TO 80                                                            COR  88     CODERD
C                                                                         COR  89     CODERD
C        FINAL PARAMETER RUN                                              COR  90     CODERD
C                                                                         COR  91     CODERD
   50 IF (ALPH(3).NE.FINE) GO TO 510                                      COR  92     CODERD
      PARFIN=.TRUE.                                                       COR  93     CODERD
      PCHGID=FINE                                                         COR  94     CODERD
      WRITE (LB3D) (FINE,I=1,50)                                          VERS   3    CODERD
      CALL INCORE (0)                                                     COR  96     CODERD
      GO TO 80                                                            COR  97     CODERD
C                                                                         COR  98     CODERD
C        CHECK FOR GENERAL PROBLEM                                        COR  99     CODERD
C                                                                         COR 100     CODERD
   60 CONTINUE                                                            COR 101     CODERD
      IF (ALPH(3).NE.GENRLP) GO TO 40                                     COR 102     CODERD
      GENERL=.TRUE.                                                       COR 103     CODERD
      DO 70 I=1,10                                                        COR 104     CODERD
      LOC(I)=0                                                            COR 105     CODERD
   70 CONTINUE                                                            COR 106     CODERD
C                                                                         COR 107     CODERD
C        SET UP TITLE                                                     COR 108     CODERD
C                                                                         COR 109     CODERD
   80 CONTINUE                                                            COR 110     CODERD
      M=0                                                                 COR 111     CODERD
      J1=1                                                                COR 112     CODERD
   90 CALL SREADC(2)                                                          V 6     CODERD
      IF (COL1.NE.COMMNT) GO TO 100                                       COR 114     CODERD
      WRITE (NOUT,640) BLANK,ALPH,COL1                                    COR 115     CODERD
      GO TO 90                                                            COR 116     CODERD
```

```
  100 CALL SREADC(3)                                                    V 6       CODERD
      IF (CODE.EQ.END) GO TO 120                                        CDR 118   CODERD
      PTITLE=.TRUE.                                                     CDR 119   CODERD
      WRITE (NOUT,630) CODE,N,(ALPH(I),I=1,N)                           CDR 120   CODERD
      M=M+N                                                             CDR 121   CODERD
      IF (ILAST.GT.20) GO TO 90                                         CDR 122   CODERD
      J=J1                                                              CDR 123   CODERD
      K=0                                                               CDR 124   CODERD
      DO 110 I=J,M                                                      CDR 125   CODERD
      ILAST=I                                                           CDR 126   CODERD
      IF (I.GT.20) GO TO 90                                             CDR 127   CODERD
      K=K+1                                                             CDR 128   CODERD
      TITLE(I)=ALPH(K)                                                  CDR 129   CODERD
  110 CONTINUE                                                          CDR 130   CODERD
      J1=M+1                                                            CDR 131   CODERD
      GO TO 90                                                          CDR 132   CODERD
  120 WRITE (NOUT,650) CODE                                             CDR 133   CODERD
      IF ((ILAST.EQ.0).AND.(PARINT.OR.PARFIN)) GO TO 140                CDR 134   CODERD
      IF (ILAST.GE.20) GO TO 140                                        CDR 135   CODERD
      ILAST=ILAST+1                                                     CDR 136   CODERD
C                                                                       CDR 137   CODERD
C        FILL OUT TITLE WITH BLANKS                                     CDR 138   CODERD
C                                                                       CDR 139   CODERD
      DO 130 I=ILAST,20                                                 CDR 140   CODERD
      TITLE(I)=BLANK                                                    CDR 141   CODERD
  130 CONTINUE                                                          CDR 142   CODERD
C                                                                       CDR 143   CODERD
C        WRITE TITLE ON TAPES                                           CDR 144   CODERD
C                                                                       CDR 145   CODERD
  140 CONTINUE                                                          CDR 146   CODERD
      CALL WRTDTA (0)                                                   CDR 147   CODERD
      IF (PARINT.OR.PARFIN) GO TO 530                                   CDR 148   CODERD
      CALL WRTPMT (0)                                                   CDR 149   CODERD
C                                                                       CDR 150   CODERD
C        ZERO ARRAY OF FIXED CONSTANTS FOR CALLS TO DATARD              CDR 151   CODERD
C                                                                       CDR 152   CODERD
      DO 150 I=1,50                                                     CDR 153   CODERD
      IFIXC(I)=0                                                        CDR 154   CODERD
  150 CONTINUE                                                          CDR 155   CODERD
      IF (.NOT.GENERL) GO TO 160                                        CDR 156   CODERD
      IFIXC(31)=2                                                       CDR 157   CODERD
      GO TO 260                                                         CDR 158   CODERD
  160 CONTINUE                                                          CDR 159   CODERD
      IF (LONG) IFIXC(31)=1                                             CDR 160   CODERD
      FIXC(50)=460.0                                                    VERS-005  CODERD
C                                                                       CDR 161   CODERD
C        READ (BCD 3NODE DATA) BLOCK                                    CDR 162   CODERD
C                                                                       CDR 163   CODERD
  170 CONTINUE                                                          CDR 164   CODERD
      CALL SREADC(1)                                                    V 6       CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 166   CODERD
      IF (COL1.EQ.COMMNT) GO TO 170                                     CDR 167   CODERD
      IF (ALPH(1).EQ.REMARK) GO TO 170                                  CDR 168   CODERD
      IF (ALPH(3).NE.TEMPB) GO TO 510                                   CDR 169   CODERD
  180 CONTINUE                                                          CDR 170   CODERD
      CALL SREADC(1)                                                    V 6       CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 172   CODERD
      IF (COL1.EQ.COMMNT) GO TO 180                                     CDR 173   CODERD
      IF (ALPH(1).EQ.END) GO TO 230                                     CDR 174   CODERD
      KBRNCH=1                                                          CDR 175   CODERD
      LNODE=.TRUE.                                                      CDR 176   CODERD
      LCOND=.FALSE.                                                     CDR 177   CODERD
      LCONST=.FALSE.                                                    CDR 178   CODERD
```

```
      LARRAY=.FALSE.                                                    CDR 179    CODERD
      NINC=LENBKT/6                                                     CDR 180    CODERD
      LOC(1)=1                                                          CDR 181    CODERD
      LEN(1)=0                                                          CDR 182    CODERD
      DO 190 I=2,6                                                      CDR 183    CODERD
      LOC(I)=LOC(I-1)+NINC                                              CDR 184    CODERD
      LEN(I)=0                                                          CDR 185    CODERD
  190 CONTINUE                                                          CDR 186    CODERD
      CALL DATARD                                                       CDR 187    CODERD
      CALL SQUEEZ (1,5)                                                 CDR 188    CODERD
      CALL WRTDTA (1)                                                   CDR 189    CODERD
      CALL WRTPMT (1)                                                   CDR 190    CODERD
      IF (.NOT.LPRINT) GO TO 230                                        CDR 191    CODERD
      WRITE (NOUT,720) NND,NNA,NNB,NNT                                  CDR 192    CODERD
      WRITE (NOUT,700) (I,LOC(I),LEN(I),I=1,5)                          CDR 193    CODERD
      M=LOC(5)+LEN(5)-1                                                 CDR 194    CODERD
      WRITE (NOUT,710) (I,IB(I),B(I),B(I),I=1,M)                        CDR 195    CODERD
      GO TO 230                                                         CDR 196    CODERD
C                                                                       CDR 197    CODERD
C        READ (BCD 3SOURCE DATA) BLOCK IF ANY                           CDR 198    CODERD
C                                                                       CDR 199    CODERD
  200 CONTINUE                                                          CDR 200    CODERD
      IF (ALPH(3).NE.QB) GO TO 510                                      CDR 201    CODERD
  210 CONTINUE                                                          CDR 202    CODERD
      CALL SREADC(1)                                                        V 6    CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 204    CODERD
      IF (COL1.EQ.COMMNT) GO TO 210                                     CDR 205    CODERD
      IF (ALPH(1).EQ.END) GO TO 230                                     CDR 206    CODERD
      LNODE=.FALSE.                                                     CDR 207    CODERD
      LQ=.TRUE.                                                         CDR 208    CODERD
      LEN(2)=0                                                          CDR 209    CODERD
      LEN(3)=0                                                          CDR 210    CODERD
      CALL DATARD                                                       CDR 211    CODERD
      CALL SQUEEZ (1,5)                                                 CDR 212    CODERD
      IF (.NOT.LPRINT) GO TO 220                                        CDR 213    CODERD
      WRITE (NOUT,700) (I,LOC(I),LEN(I),I=1,4)                          CDR 214    CODERD
      M1=LOC(2)                                                         CDR 215    CODERD
      M=LOC(3)+LEN(3)-1                                                 CDR 216    CODERD
      WRITE (NOUT,710) (I,B(I),B(I),B(I),I=M1,M)                        CDR 217    CODERD
  220 CONTINUE                                                          CDR 218    CODERD
C                                                                       CDR 219    CODERD
C        READ (BCD 3CONDUCTOR DATA) BLOCK                               CDR 220    CODERD
C                                                                       CDR 221    CODERD
  230 CONTINUE                                                          CDR 222    CODERD
      JPSTOT=0                                                          VERS 7     CODERD
      END FILE 27                                                                  CODERD
      REWIND 27                                                                    CODERD
      CALL SREADC(1)                                                        V 6    CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 224    CODERD
      IF (COL1.EQ.COMMNT) GO TO 230                                     CDR 225    CODERD
      IF (ALPH(1).EQ.REMARK) GO TO 230                                  CDR 226    CODERD
      IF (ALPH(3).NE.CONDB) GO TO 200                                   CDR 227    CODERD
  240 CONTINUE                                                          CDR 228    CODERD
      CALL SREADC(1)                                                        V 6    CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 230    CODERD
      IF (COL1.EQ.COMMNT) GO TO 240                                     CDR 231    CODERD
      IF (ALPH(1).EQ.END) GO TO 260                                     CDR 232    CODERD
      KBRNCH=2                                                          CDR 233    CODERD
      LNODE=.FALSE.                                                     CDR 234    CODERD
      LCOND=.TRUE.                                                      CDR 235    CODERD
      LCONST=.FALSE.                                                    CDR 236    CODERD
      LARRAY=.FALSE.                                                    CDR 237    CODERD
      NNEW=LENBKT-(LOC(5)+LEN(5))+1                                     CDR 238    CODERD
```

```
      NINC=NNEW/5                                        COR 239        CODERD
      LOC(6)=LOC(5)+LEN(5)                               COR 240        CODERD
      LEN(6)=0                                           COR 241        CODERD
      DO 250 I=7,10                                      COR 242        CODERD
      LOC(I)=LOC(I-1)+NINC                               COR 243        CODERD
      LEN(I)=0                                           COR 244        CODERD
  250 CONTINUE                                           COR 245        CODERD
      CALL DATARD                                        COR 246        CODERD
      JJIST=LOC(6)                                       VERS 7         CODERD
      JJEND=LOC(6)+LEN(6)-1                              VERS 7         CODERD
      WRITE (27) (B(JJ),JJ=JJIST,JJEND)                  VERS 7         CODERD
      CALL WRIPMT(2)                                     VERS 7         CODERD
      CALL WRTDTA(2)                                     VERS 7         CODERD
      READ (27) (B(JJ),JJ=JJIST,JJEND)                   VERS 7         CODERD
      CALL SQUEEZ(6,10)                                  VERS 7         CODERD
      IF (.NOT.LPRINT) GO TO 260                         COR 250        CODERD
      WRITE (NOUT,730) NGL,NGR,NGT                       COR 251        CODERD
      WRITE (NOUT,700) (I,LOC(I),LEN(I),I=6,10)          COR 252        CODERD
      M1=LOC(6)                                          COR 253        CODERD
      M=LOC(10)+LEN(10)-1                                COR 254        CODERD
      WRITE (NOUT,710) (I,IB(I),B(I),B(I),I=M1,M)        COR 255        CODERD
C                                                        COR 256        CODERD
C        READ (BCD 3CONSTANTS DATA) BLOCK                COR 257        CODERD
C                                                        COR 258        CODERD
  260 CONTINUE                                           COR 259        CODERD
      CALL SREADC(1)                                         V 6        CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1             COR 261        CODERD
      IF (COL1.EQ.COMMNT) GO TO 260                      COR 262        CODERD
      IF (ALPH(1).EQ.REMARK) GO TO 260                   COR 263        CODERD
      IF (ALPH(3).NE.CONSTB) GO TO 510                   COR 264        CODERD
      IF (ALPH(14).EQ.PRINT) KTPRNT=.TRUE.               COR 265        CODERD
  270 CONTINUE                                           COR 266        CODERD
      CALL SREADC(1)                                         V 6        CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1             COR 268        CODERD
      IF (COL1.EQ.COMMNT) GO TO 270                      COR 269        CODERD
      KBRNCH=3                                           COR 270        CODERD
      LNODE=.FALSE.                                      COR 271        CODERD
      LCOND=.FALSE.                                      COR 272        CODERD
      LCONST=.TRUE.                                      COR 273        CODERD
      LARRAY=.FALSE.                                     COR 274        CODERD
      NNEW=LENBKT-(LOC(10)+LEN(10))+1                    COR 275        CODERD
      NINC=NNEW/2                                        COR 276        CODERD
      LOC(11)=LOC(10)+LEN(10)                            COR 277        CODERD
      IF (GENERL) LOC(11)=1                                             CODERD
      LEN(11)=0                                          COR 278        CODERD
      LOC(12)=LOC(11)+NINC                               COR 279        CODERD
      LEN(12)=0                                          COR 280        CODERD
      CALL DATARD                                        COR 281        CODERD
      CALL SQUEEZ (11,12)                                COR 282        CODERD
      CALL WRTDTA (3)                                    COR 283        CODERD
      CALL WRIPMT (3)                                    COR 284        CODERD
      IF (.NOT.LPRINT) GO TO 280                         COR 285        CODERD
      WRITE (NOUT,740) NUC,NEC1,NEC2,NCT                 COR 286        CODERD
      WRITE (NOUT,750) (I,IFIXC(I),IFIXC(I),IFIXC(I),I=1,50)  COR 287   CODERD
      WRITE (NOUT,700) (I,LOC(I),LEN(I),I=11,12)         COR 288        CODERD
      M1=LOC(11)                                         COR 289        CODERD
      M=LOC(12)+LEN(12)-1                                COR 290        CODERD
      WRITE (NOUT,710) (I,IB(I),B(I),B(I),I=M1,M)        COR 291        CODERD
C                                                        COR 292        CODERD
C        READ (BCD 3ARRAY DATA) BLOCK                    COR 293        CODERD
C                                                        COR 294        CODERD
  280 CONTINUE                                           COR 295        CODERD
      CALL SREADC(1)                                         V 6        CODERD
```

```
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                          CDR 297   CODERD
      IF (COL1.EQ.COMMNT) GO TO 290                                   CDR 298   CODERD
      IF (ALPH(1).EQ.REMARK) GO TO 280                                CDR 299   CODERD
      IF (ALPH(3).NE.ARRYB) GO TO 510                                 CDR 300   CODERD
      IF (ALPH(14).EQ.PRINT) AYPRNT=.TRUE.                            CDR 301   CODERD
  290 CONTINUE                                                        CDR 302   CODERD
      CALL SREADC(1)                                                      V 6   CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                          CDR 304   CODERD
      IF (COL1.EQ.COMMNT) GO TO 290                                   CDR 305   CODERD
      IF (ALPH(1).EQ.END) GO TO 300                                   CDR 306   CODERD
      KBRNCH=4                                                        CDR 307   CODERD
      LNODE=.FALSE.                                                   CDR 308   CODERD
      LCOND=.FALSE.                                                   CDR 309   CODERD
      LCONST=.FALSE.                                                  CDR 310   CODERD
      LARRAY=.TRUE.                                                   CDR 311   CODERD
      LOC(13)=LOC(12)+LEN(12)                                         CDR 312   CODERD
      LEN(13)=0                                                       CDR 313   CODERD
      LOC(14)=LOC(13)+200                                             CDR 314   CODERD
      LEN(14)=0                                                       CDR 315   CODERD
      LOC(15)=LOC(14)+200                                             CDR 316   CODERD
      LEN(15)=0                                                       CDR 317   CODERD
      CALL DATARD                                                     CDR 318   CODERD
      CALL IMBED                                                          V 9   CODERD
  300 CONTINUE                                                        CDR 319   CODERD
      CALL SQUEEZ (13,15)                                             CDR 320   CODERD
      CALL WRTDTA (4)                                                 CDR 321   CODERD
      CALL WRTPMT (4)                                                 CDR 322   CODERD
      IF (.NOT.LPRINT) GO TO 310                                      CDR 323   CODERD
      WRITE (NOUT,760) LENA                                           CDR 324   CODERD
      WRITE (NOUT,700) (I,LOC(I),LEN(I),I=13,15)                      CDR 325   CODERD
      M1=LOC(13)                                                      CDR 326   CODERD
      M=LOC(15)+LEN(15)-1                                             CDR 327   CODERD
      WRITE (NOUT,710) (I,IB(I),B(I),B(I),I=M1,M)                     CDR 328   CODERD
C                                                                     CDR 329   CODERD
C        NORMAL RETURN                                                CDR 330   CODERD
C                                                                     CDR 331   CODERD
  310 CONTINUE                                                        CDR 332   CODERD
      IF (GENERL) GO TO 320                                           CDR 333   CODERD
C        SET LOC AND LEN FOR CALL TO PSEUDO                           CDR 334   CODERD
      NNEW=LENBKT-(LOC(15)+LEN(15))+1                                 CDR 335   CODERD
      NINC=NNEW/2                                                     CDR 336   CODERD
      LOC(16)=LOC(15)+LEN(15)                                         CDR 337   CODERD
      LEN(16)=0                                                       CDR 338   CODERD
      LOC(17)=LOC(16)+NINC                                            CDR 339   CODERD
      LEN(17)=0                                                       CDR 340   CODERD
C        CONVERT ARRAYS AND CONSTANTS TO FORTRAN                      CDR 341   CODERD
  320 CONTINUE                                                        CDR 342   CODERD
      M=4                                                             CDR 343   CODERD
      J=1                                                             CDR 344   CODERD
  330 IF (LEN(M).EQ.0) GO TO 400                                      CDR 345   CODERD
      K=LOC(M)                                                        CDR 346   CODERD
      KEND=K+LEN(M)                                                   CDR 347   CODERD
  340 ITYPE=FLD(0,6,IB(K))                                            CDR 348   CODERD
      FLD(0,5,IADDR)=ITYPE                                            CDR 349   CODERD
      ITYPE=MOD(ITYPE,4)                                              CDR 350   CODERD
      IEND=1                                                          CDR 351   CODERD
      IF ((ITYPE.EQ.0).OR.(ITYPE.EQ.2)) IEND=2                        CDR 352   CODERD
      DO 390 I=1,IEND                                                 CDR 353   CODERD
      LITA=FLD(6,1,IB(K))                                             CDR 354   CODERD
      IANUM=FLD(7,14,IB(K))                                           CDR 355   CODERD
      IF (LITA.EQ.1) GO TO 350                                        CDR 356   CODERD
      CALL RELACT (2,IANUM,J,M)                                       CDR 357   CODERD
      GO TO 360                                                       CDR 358   CODERD
```

```
  350 IANUM=IANUM+NUC+1                                    CDR 359   CODERO
      IF (M.EQ.9) IANUM=IANUM+NEC1                         CDR 360   CODERO
  360 FLD(5,1,IADDR)=LITA                                  CDR 361   CODERO
      FLD(6,16,IADDR)=IANUM                                CDR 362   CODERO
      LITK=FLD(21,1,IB(K))                                 CDR 363   CODERO
      KNUM=FLD(22,14,IB(K))                                CDR 364   CODERO
      IF (LITK.EQ.1) GO TO 370                             CDR 365   CODERO
      CALL RELACT (3,KNUM,J,M)                             CDR 366   CODERO
      GO TO 380                                            CDR 367   CODERO
  370 KNUM=KNUM+NUC+1                                      CDR 368   CODERO
      IF (M.EQ.9) KNUM=KNUM+NEC1                           CDR 369   CODERO
      IF (KNUM.GT.8191) CALL ERRMES (40,KNUM,0,0)          CDR 370   CODERO
  380 FLD(22,1,IADDR)=LITK                                 CDR 371   CODERO
      FLD(23,13,IADDR)=KNUM                                CDR 372   CODERO
      IB(K)=IADDR                                          CDR 373   CODERO
      IADDR=0                                              CDR 374   CODERO
      K=K+1                                                CDR 375   CODERO
  390 CONTINUE                                             CDR 376   CODERO
      J=J+1                                                CDR 377   CODERO
      IF ((ITYPE.EQ.0).OR.(ITYPE.EQ.3)) K=K+1              CDR 378   CODERO
      IF (K.LT.KEND) GO TO 340                             CDR 379   CODERO
  400 IF (M.EQ.9) GO TO 410                                CDR 380   CODERO
      M=9                                                  CDR 381   CODERO
      GO TO 330                                            CDR 382   CODERO
  410 CONTINUE                                             CDR 383   CODERO
      IF (.NOT.LO) GO TO 490                               CDR 384   CODERO
      NCC=NUC+NEC1+NEC2+1                                  CDR 385   CODERO
      J=-1                                                 CDR 386   CODERO
      K=LOC(2)                                             CDR 387   CODERO
      KEND=K+LEN(2)                                        CDR 388   CODERO
      IADDR=0                                              CDR 389   CODERO
  420 ITYPE=FLD(0,6,IB(K))                                 CDR 390   CODERO
      K=K+1                                                CDR 391   CODERO
      IEND=1                                               CDR 392   CODERO
      IF (ITYPE.GT.3) IEND=2                               CDR 393   CODERO
      DO 480 I=1,IEND                                      CDR 394   CODERO
      IF (ITYPE.EQ.1) GO TO 450                            CDR 395   CODERO
      LITA=FLD(6,1,IB(K))                                  CDR 396   CODERO
      IANUM=FLD(7,14,IB(K))                                CDR 397   CODERO
      IF (LITA.EQ.1) GO TO 430                             CDR 398   CODERO
      CALL RELACT (2,IANUM,J,2)                            CDR 399   CODERO
      GO TO 440                                            CDR 400   CODERO
  430 IANUM=IANUM+NCC                                      CDR 401   CODERO
  440 FLD(5,1,IADDR)=LITA                                  CDR 402   CODERO
      FLD(6,16,IADDR)=IANUM                                CDR 403   CODERO
  450 LITK=FLD(21,1,IB(K))                                 CDR 404   CODERO
      KNUM=FLD(22,14,IB(K))                                CDR 405   CODERO
      IF (LITK.EQ.1) GO TO 460                             CDR 406   CODERO
      CALL RELACT (3,KNUM,J,2)                             CDR 407   CODERO
      GO TO 470                                            CDR 408   CODERO
  460 KNUM=KNUM+NCC                                        CDR 409   CODERO
      IF (KNUM.GT.8191) CALL ERRMES (40,KNUM,0,0)          CDR 410   CODERO
  470 FLD(22,1,IADDR)=LITK                                 CDR 411   CODERO
      FLD(23,13,IADDR)=KNUM                                CDR 412   CODERO
      IB(K)=IADDR                                          CDR 413   CODERO
      IADDR=0                                              CDR 414   CODERO
      K=K+1                                                CDR 415   CODERO
  480 CONTINUE                                             CDR 416   CODERO
      J=J-1                                                CDR 417   CODERO
      IF (K.LT.KEND) GO TO 420                             CDR 418   CODERO
  490 CONTINUE                                             CDR 419   CODERO
      LNODE=.FALSE.                                        CDR 420   CODERO
      LCOND=.FALSE.                                        CDR 421   CODERO
```

```
      LCONST=.FALSE.                                                    CDR 422         CODERD
      LARRAY=.FALSE.                                                    CDR 423         CODERD
      RETURN                                                            CDR 424         CODERD
C                                                                       CDR 425         CODERD
C        ERROR RETURN                                                   CDR 426         CODERD
C                                                                       CDR 427         CODERD
  500 WRITE (NOUT,690)                                                  CDR 428         CODERD
      ERDATA=2.0                                                        CDR 429         CODERD
      GO TO 520                                                         CDR 430         CODERD
  510 WRITE (NOUT,680)                                                  CDR 431         CODERD
      ERDATA=2.0                                                        CDR 432         CODERD
  520 CONTINUE                                                          CDR 433         CODERD
      ENDRUN=1.0                                                        CDR 434         CODERD
      RETURN                                                            CDR 435         CODERD
C                                                                       CDR 436         CODERD
C        PARAMETER RUNS                                                 CDR 437         CODERD
C                                                                       CDR 438         CODERD
  530 CONTINUE                                                          CDR 439         CODERD
      LEND=.FALSE.                                                      CDR 440         CODERD
      NOREAD=.FALSE.                                                    CDR 441         CODERD
      IST=1                                                             CDR 442         CODERD
      IF (GENERL) IST=3                                                 CDR 443         CODERD
      DO 610 I=IST,4                                                    CDR 444         CODERD
      IF (LEND.OR.NOREAD) GO TO 580                                     CDR 445         CODERD
  540 CALL SREADC(1)                                                        V 6         CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 447         CODERD
      IF (COL1.EQ.COMMNT) GO TO 540                                     CDR 448         CODERD
      IF (ALPH(1).EQ.REMARK) GO TO 540                                  CDR 449         CODERD
      IF (ALPH(3).NE.ENDPRM) GO TO 550                                  CDR 450         CODERD
      LEND=.TRUE.                                                       CDR 451         CODERD
      GO TO 580                                                         CDR 452         CODERD
  550 IBHC=ALPH(3)                                                      CDR 453         CODERD
      DO 560 J=1,4                                                      CDR 454         CODERD
      IF (IBHC.EQ.BLOCK(J)) GO TO 570                                   CDR 455         CODERD
  560 CONTINUE                                                          CDR 456         CODERD
      GO TO 510                                                         CDR 457         CODERD
  570 CALL SREADC(1)                                                        V 6         CODERD
      WRITE (NOUT,670) BLANK,COL27,ALPH,COL1                            CDR 459         CODERD
      IF (COL1.EQ.COMMNT) GO TO 570                                     CDR 460         CODERD
      IF (ALPH(3).EQ.REMARK) GO TO 570                                  CDR 461         CODERD
  580 CALL INCORE (I)                                                   CDR 462         CODERD
      IF (ALPH(1).EQ.END) GO TO 600                                     CDR 463         CODERD
      NOREAD=.TRUE.                                                     CDR 464         CODERD
      IF (IBHC.NE.BLOCK(I)) GO TO 600                                   CDR 465         CODERD
      NOREAD=.FALSE.                                                    CDR 466         CODERD
      KBRNCH=I                                                          CDR 467         CODERD
      DO 590 J=1,4                                                      CDR 468         CODERD
      LLOGIC(J)=.FALSE.                                                 CDR 469         CODERD
  590 CONTINUE                                                          CDR 470         CODERD
      LLOGIC(I)=.TRUE.                                                  CDR 471         CODERD
      CALL DATARD                                                       CDR 472         CODERD
  600 CALL WRTDTA (I)                                                   CDR 473         CODERD
  610 CONTINUE                                                          CDR 474         CODERD
      GO TO 490                                                         CDR 475         CODERD
C                                                                       CDR 476         CODERD
C                                                                       CDR 477         CODERD
  620 FORMAT (1H1//)                                                    CDR 478         CODERD
  630 FORMAT (7X,A4,I1,11A6,A2)                                         CDR 479         CODERD
  640 FORMAT (A1,13A6,2A1)                                              CDR 480         CODERD
  650 FORMAT (7X,A6)                                                    CDR 481         CODERD
  660 FORMAT (7X,A4,A1,11A6,A2)                                         CDR 482         CODERD
  670 FORMAT (A1,A6,A4,A1,11A6,A2,A1)                                   CDR 483         CODERD
  680 FORMAT (6H * * *,82H DATA  BLOCKS  IN  IMPROPER  ORDER  OR  ILLEGACDR 484         CODERD
```

```
     1L  BLOCK  DESIGNATION  ENCOUNTERED .)                          CDR 485          CODERD
  690 FORMAT (6H * * *,90H THE PSEUDO COMPUTE SEQUENCE INDICATOR MUST BECDR 486          CODERD
     1 EITHER SPCS OR LPCS, AND START IN COLUMN 21)                  CDR 487          CODERD
  700 FORMAT (19H ARRAYS LOC AND LEN,/(3I10))                        CDR 488          CODERD
  710 FORMAT (12H DATA BUCKET,/(I10,I20,E20.5,5X,O20))               CDR 489          CODERD
  720 FORMAT (/4H NND,I6,4H NNA,I6,4H NNB,I6,4H NNT,I6)              CDR 490          CODERD
  730 FORMAT (/4H NGL,I6,4H NGR,I6,4H NGT,I6)                        CDR 491          CODERD
  740 FORMAT (/4H NUC,I6,5H NEC1,I6,5H NEC2,I6,4H NCT,I6)            CDR 492          CODERD
  750 FORMAT (/22H FIXED CONSTANTS ARRAY,/(I3,I20,E20.5,8X,O12))     CDR 493          CODERD
  760 FORMAT (/5H LENA,I6)                                           CDR 494          CODERD
      END                                                            CDR 495-         CODERD
```

```
      SUBROUTINE IMBED                                          IMBED
C                                                               IMBED
      COMMON /BUCKET/ IB(1)                                     IMBED
      COMMON /DATA  / DUM1(6), NGT, NUC, DUM2(4), ERDATA  *NEW  IMBED
      COMMON /POINT / LOC(20), LEN(20)                    **-1  IMBED
      DIMENSION KEYA(4)/2H*A,2H*X,2H*T,2H*G/                    IMBED
      LOGICAL CRDERR                                            IMBED
C                                                               IMBED
      L1 = LOC(13)                                              IMBED
      L2 = LEN(13) + L1 -1                                      IMBED
      MM = LOC(14)                                              IMBED
      M2 = LOC(15) - 1                                          IMBED
      DO 500 M=L1,L2                                            IMBED
      M1 = M2 + 1                                               IMBED
      M2 = M2 + IB(MM)                                          IMBED
      MM = MM + 1                                               IMBED
      KEY = 6H                                                  IMBED
      DO 400 I=M1,M2                                            IMBED
      FLD(0,12,KEY) = FLD(0,12,IB(I))                           IMBED
      DO 10 K=1,4                                               IMBED
      IF(KEY .EQ. KEYA(K)) GO TO 40                             IMBED
   10 CONTINUE                                                  IMBED
      GO TO 400                                                 IMBED
   40 NUM = IB(I)                                               IMBED
      CALL CONVRT(12,30,NUM,CRDERR)                             IMBED
      IF (CRDERR) GO TO 380                                     IMBED
C                                                               IMBED
   85 GO TO (100,200,300,350), K                                IMBED
C                                                               IMBED
C ARRAYS                                                        IMBED
C                                                               IMBED
  100 L= 1                                                      IMBED
      LL = LOC(14)                                              IMBED
      IST = LOC(13)                                             IMBED
      IEND = IST + LEN(13) - 1                                  IMBED
      DO 140 JJ=IST,IEND                                        IMBED
      IF(NUM .EQ. IB(JJ)) GO TO 390                             IMBED
      L = L + IB(LL)                                            IMBED
      LL = LL + 1                                               IMBED
  140 CONTINUE                                                  IMBED
      GO TO 380                                                 IMBED
C                                                               IMBED
C CONSTANTS                                                     IMBED
C                                                               IMBED
  200 NLOC = LOC(11)                                            IMBED
      NLEN = NUC                                                IMBED
      GO TO 360                                                 IMBED
C                                                               IMBED
C TEMPERATURES                                                  IMBED
C                                                               IMBED
  300 NLOC = LOC(1)                                             IMBED
      NLEN = LEN(1)                                             IMBED
      GO TO 360                                                 IMBED
C                                                               IMBED
C CONDUCTORS                                                    IMBED
C                                                               IMBED
  350 NLOC = LOC(5)                                             IMBED
      NLEN = NGT                                                IMBED
C                                                               IMBED
```

```
  360 CALL SEARCH(NUM,IB(NLOC),NLEN,L)                                      IMBED
      IF(L) 380,380,390                                                     IMBED
  380 ERDATA = 1.0                                                          IMBED
      NN = I - M1 + 1                                                       IMBED
      WRITE(6,385) IB(I),NN,IB(M)                                           IMBED
  385 FORMAT(8H * * *   A6, 23H REFERENCED AT LOCATION  I5,                 IMBED
     1 9H OF ARRAY  I5, 26H IS NOT IN THE LIST  * * *)                      IMBED
      GO TO 400                                                             IMBED
C                                                                           IMBED
  390 IB(I) = L                                                             IMBED
C                                                                           IMBED
  400 CONTINUE                                                              IMBED
C                                                                           IMBED
  500 CONTINUE                                                              IMBED
      RETURN                                                                IMBED
C                                                                           IMBED
      END                                                                   IMBED
```

```
SINDA    SINDA    SINDA    SINDA    SINDA    SINDA    SINDA    SINDA    SINDA    SINDA    SINDA

 1.    MLINK    SEG    PREPRO-BLKCRD-WRTBLK-STFFB-FINDRM-SQUEEZ-SREADC                     SINDA
 2.    D        SEG    MLINK-*(EDIT,MUR,GENLNK,A,B-BB,C,F)                                   SINDA
 3.    A        SEG      PSEUDO-PCS2                                                         SINDA
 4.    B        SEG      CODERD-DATARD-ERRMES-CONVRT-TYPCHK-QDATA                            SINDA
 5.    BB  SEG  RELACT-WRTDTA-WRTPMT-INCORE-SETFMT-GENUX-*(NODES,CONDS)           **-1      SINDA
 6.    C        SEG      PRESUB-SINDA4-MXTOFN-ALPINT                                         SINDA
 7.    F        SEG      SPLIT-SKIP                                                          SINDA
 8.             BLK    BUFBLK,MODNEW,MURCRD                                                  SINDA
 9.    MUR      SEG    MODUR-BUFBLK-MODNEW-MURCRD                                            SINDA
```

LISTING OF FILE 2 OF TAPE A19506

ABS1 ABS1 ABS1 ABS1 ABS1 ABS1 ABS1 ABS1 ABS1 ABS1 ABS1

```
      FUNCTION ABS1(X)                                    ABS1
      ABS1 = ABS(X)                                       ABS1
      RETURN                                              ABS1
      END                                                 ABS1
```

```
      SUBROUTINE CABIN(NLOC,IC,SUMQL,SUMWL)                          CABIN
C                                                                    CABIN
      LOGICAL EXPLCT                                                 CABIN
C                                                                    CABIN
      COMMON /ARRAY / DATA(1)                                        CABIN
      COMMON /FIXCON/ CON(1)                                         CABIN
      COMMON /TEMP  / T(1)                                           CABIN
      COMMON /SOURCE/ Q(1)                                           CABIN
      COMMON /XSPACE/ NDIM, NTH, EXT(1)                              CABIN
      COMMON /DIMENS/ NND, NNA                                       CABIN
C                                                                    CABIN
      DIMENSION NLOC(1)                                              CABIN
      DIMENSION NDATA(1)                                             CABIN
      DIMENSION NEXT(1)                                        *NEW  CABIN
C                                                                    CABIN
      EQUIVALENCE (CON(1),TIME), (CON(2),TINC), (CON(22),DTIMEI)     CABIN
      EQUIVALENCE (DATA,NDATA), (EXT,NEXT)                     *NEW  CABIN
C                                                              **-1  CABIN
      DEFINE DTAU(I) = EXT(NNC+I)                                    CABIN
C                                                                    CABIN
      NNT = NNA + NND                                          **-1  CABIN
      NNC = NTH - NNT                                          **-1  CABIN
      EXPLCT = .TRUE.                                                CABIN
      IF(DTIMEI .GT. 0.0) EXPLCT = .FALSE.                           CABIN
      IF(NLOC(1) .EQ. 6) GO TO 102                                   CABIN
      CALL TOPLIN                                                    CABIN
      WRITE(6,101) NLOC(1)                                           CABIN
  101 FORMAT(57H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CABIN, IC  CABIN
     1= I5, 7H  * * *)                                               CABIN
      CALL WLKBCK                                                    CABIN
      CALL EXIT                                                      CABIN
C                                                                    CABIN
  102 NST  = NLOC(2)                                                 CABIN
      NCRV = NLOC(3)                                                 CABIN
      NCON = NLOC(4)                                                 CABIN
      LHC  = NLOC(5)                                                 CABIN
      LHFP = NLOC(6)                                                 CABIN
      LHTB = NLOC(7)                                                 CABIN
      LAR  = NTH + 1                                                 CABIN
C                                                                    CABIN
      NSPT = 0                                                       CABIN
      NL1 = 0                                                        CABIN
      NL2 = 0                                                        CABIN
      NL3 = 0                                                        CABIN
      IF(LHTB .GT. 0) NL1 = NDATA(LHTB)                              CABIN
      IF(LHFP .GT. 0) NL2 = NDATA(LHFP)                              CABIN
      IF(LHC  .GT. 0) NL3 = NDATA(LHC )                              CABIN
      NSPT = (NL1/4 + NL2/5 + NL3/2) * 3                             CABIN
      NEXT(LAR) = NSPT                                         *NEW  CABIN
      IF(NDIM .GE. NSPT) GO TO 104                             **-1  CABIN
      NEED = NSPT - NDIM                                             CABIN
      CALL TOPLIN                                                    CABIN
      WRITE(6,103) NEED                                              CABIN
  103 FORMAT(83H0* * *  INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR CABIN  CABIN
     1 ANALYSIS SUBROUTINE  * * * // 8X 5HSHORT I5, 10H LOCATIONS)    CABIN
      CALL WLKBCK                                                    CABIN
      CALL EXIT                                                      CABIN
C                                                                    CABIN
  104 CONTINUE                                                       CABIN
```

```
C                                                                   CABIN
      NS = NDATA(NST+1)                                             CABIN
      NV1 = IABS(NDATA(NST)) - 1                                    CABIN
      IF (NDATA(NST) .LT. 0) GO TO 2                                CABIN
      IF(NS .NE. NV1/3) CALL ERR(3H1ST)                             CABIN
      NDATA(NST) = -NDATA(NST)                                      CABIN
    2 IF (NDATA(NCRV) .LT. 0) GO TO 4                               CABIN
      IF(NDATA(NCRV) .NE. 8) CALL ERR(3H2ND)                        CABIN
      NDATA(NCRV) = -NDATA(NCRV)                                    CABIN
    4 IF(NDATA(NCON) .NE. 11) CALL ERR(3H3RD)                       CABIN
      NCPA = NDATA(NCRV+4)                                          CABIN
      NCPV = NDATA(NCRV+5)                                          CABIN
      LAMDA = NDATA(NCRV+8)                                         CABIN
      RA = DATA(NCON+1)                                             CABIN
      RV = DATA(NCON+2)                                             CABIN
      VC = DATA(NCON+3)                                             CABIN
      PC = DATA(NCON+4)                                             CABIN
      XC = DATA(NCON+5)                                             CABIN
      WV = DATA(NCON+6)                                             CABIN
      PSICAB = DATA(NCON+7)                                         CABIN
      PO = DATA(NCON+8)                                             CABIN
      TO = DATA(NCON+9)                                             CABIN
      CONV = DATA(NCON+10)                                          CABIN
      TZ = DATA(NCON+11)                                            CABIN
      FLOIN = 0.0                                                   CABIN
      PSIIN = 0.0                                                   CABIN
      TIN = 0.0                                                     CABIN
      FLOCP = 0.0                                                   CABIN
      DO 5 I=1,NV1,3                                                CABIN
      LOC = NST + I + 1                                             CABIN
      LOC1 = LOC + 1                                                CABIN
      LOC2 = LOC + 2                                                CABIN
      FLO = DATA(LOC)                                               CABIN
      IF(IABS(NDATA(LOC)).LE. 99999 .AND. IABS(NDATA(LOC)).GT. 0)   CABIN
     X  FLO = POL(NDATA(LOC),TIME)                                  CABIN
      PSI = DATA(LOC1)                                              CABIN
      IF(IABS(NDATA(LOC1)).LE. 99999 .AND. IABS(NDATA(LOC1)).GT. 0) CABIN
     X  PSI = POL(NDATA(LOC1),TIME)                                 CABIN
      TEMP = DATA(LOC2)                                             CABIN
      IF(IABS(NDATA(LOC2)).LE. 99999 .AND. IABS(NDATA(LOC2)).GT. 0) CABIN
     X  TEMP = POL(NDATA(LOC2),TIME)                                CABIN
      FLOIN = FLOIN + FLO                                           CABIN
      PSIIN = PSIIN + FLO*PSI                                       CABIN
      CPIN  = (POL(NCPA,TEMP)+PSI*POL(NCPV,TEMP))/(1.0+PSI)         CABIN
      TIN   = TIN + FLO*CPIN*TEMP                                   CABIN
      FLOCP = FLOCP + FLO*CPIN                                      CABIN
    5 CONTINUE                                                      CABIN
      PSIIN = PSIIN/FLOIN                                           CABIN
      TIN = TIN/FLOCP                                               CABIN
      FLOC = POL(NDATA(NCRV+1),TIME)                                CABIN
      WVIN = TINC*FLOIN*PSIIN/(1.0+PSIIN)                           CABIN
      RG = RA*(1.0+PSIIN/XC)/(1.0+PSIIN)                            CABIN
      RHOIN = PC/(RG*(TIN-TZ))                                      CABIN
      FLOUT = FLOIN*(WV+WV/PSICAB)/VC/RHOIN                         CABIN
      WVOUT = TINC*FLOUT*PSICAB/(1.0+PSICAB)                        CABIN
      WV = WV + WVIN - WVOUT - SUMWL                                CABIN
      DATA(NCON+6) = WV                                             CABIN
      PV = WV*RV*(TC-TZ)/VC                                         CABIN
      PA = PC - PV                                                  CABIN
      WA = VC*PA/RA/(TC-TZ)                                         CABIN
      PSICAB = WV/WA                                                CABIN
      DATA(NCON+7) = PSICAB                                         CABIN
      UA = POL(NDATA(NCRV+2),TC)                                    CABIN
```

```
      UV = POL(NDATA(NCRV+3),TC)                                          CABIN
      CPA = POL(NCPA,TC)                                                  CABIN
      CPV = POL(NCPV,TC)                                                  CABIN
      CA = POL(NDATA(NCRV+6),TC)                                          CABIN
      CV = POL(NDATA(NCRV+7),TC)                                          CABIN
      UC = (XC*UA+PSICAB*UV)/(XC+PSICAB)                                  CABIN
      CPC = (CPA+PSICAB*CPV)/(1.0+PSICAB)                                 CABIN
      CC = (XC*CA+PSICAB*CV)/(XC+PSICAB)                                  CABIN
      RHOC = (WV+WA)/VC                                                   CABIN
      TC = TC + (FLOIN*(TIN-TC) - SUMQL/CPC)/(WV+WA)*TINC                 CABIN
      PRC = CC/(RHOC*CPC)                                                 CABIN
      SUMQL = 0.0                                                         CABIN
      SUMWL = 0.0                                                         CABIN
      LL = LAR                                                            CABIN
C                                                                         CABIN
      IF(LHTB .EQ. 0) GO TO 25                                            CABIN
      PRC31 = PRC**.31                                                    CABIN
      IF(MOD(NDATA(LHTB),4) .NE. 0) CALL ERR(3H7TH)                       CABIN
      DO 20 I=1,NL1,4                                                     CABIN
      LOC = LHTB + I                                                      CABIN
      J = NDATA(LOC)                                                      CABIN
      DI = DATA(LOC+1)                                                    CABIN
      AI = DATA(LOC+2)                                                    CABIN
      VIWO = DATA(LOC+3)                                                  CABIN
      VI = VIWO*FLOC                                                      CABIN
      RE = VI*DI*RHOC/UC                                                  CABIN
      IF(IFIX((RE-22000)/18000)) ,7,10                                    CABIN
      XNU = .43 + .533*SQRT(RE)*PRC31                                     CABIN
      GO TO 15                                                            CABIN
    7 XNU = .43 + .193*RE**.618*PRC31                                     CABIN
      GO TO 15                                                            CABIN
   10 XNU = .43 + .0265*RE**.805*PRC31                                    CABIN
   15 HA = AI*CC*XNU/DI                                                   CABIN
      CALL QSUM                                                           CABIN
   20 CONTINUE                                                            CABIN
C                                                                         CABIN
   25 IF(LHFP .EQ. 0) GO TO 35                                            CABIN
      PRC33 = CBRT(PRC)                                                   CABIN
      IF(MOD(NDATA(LHFP),5) .NE. 0) CALL ERR(3H6TH)                       CABIN
      DO 30 I=1,NL2,5                                                     CABIN
      LOC = LHFP + I                                                      CABIN
      J = NDATA(LOC)                                                      CABIN
      XX = DATA(LOC+1)                                                    CABIN
      XI = DATA(LOC+2)                                                    CABIN
      AI = DATA(LOC+3)                                                    CABIN
      VIWO = DATA(LOC+4)                                                  CABIN
      VI = VIWO*FLOC                                                      CABIN
      VRU = VI*RHOC/UC                                                    CABIN
      XNU = .664*PRC33*(SQRT(VRU*(XX+XI)) - SQRT(VRU*XX))                 CABIN
      HA = AI*CC*XNU/XI                                                   CABIN
      CALL QSUM                                                           CABIN
   30 CONTINUE                                                            CABIN
C                                                                         CABIN
   35 IF(LHC .EQ. 0) GO TO 45                                             CABIN
      IF(MOD(NDATA(LHC),2) .NE. 0) CALL ERR(3H5TH)                        CABIN
      DO 40 I=1,NL3,2                                                     CABIN
      LOC = LHC + I                                                       CABIN
      J = NDATA(LOC)                                                      CABIN
      HA = DATA(LOC+1)                                                    CABIN
      CALL QSUM                                                           CABIN
   40 CONTINUE                                                            CABIN
C                                                                         CABIN
   45 WVPRME = WV + WVIN - WVOUT - SUMWL                                  CABIN
```

```
      PVPRME = WVPRME*RV*(TC-TZ)/VC                                       CABIN
      SUMWL = 0.                                                          CABIN
      KK = NEXT(LAR)/3*2 + LAR                                 *NEW       CABIN
      LL = LAR                                                 **-1       CABIN
      IF(LHTB .GT. 0) CALL CONCK(NL1,4,LHTB)                              CABIN
      IF(LHFP .GT. 0) CALL CONCK(NL2,5,LHFP)                              CABIN
      IF(LHC  .GT. 0) CALL CONCK(NL3,2,LHC)                               CABIN
      RETURN                                                   **-1       CABIN
C                                                                         CABIN
      SUBROUTINE CONCK(NN,NUM,IND)                                        CABIN
      DO 60 I=1,NN,NUM                                                    CABIN
      J = NDATA(IND+I)                                                    CABIN
      LL = LL + 1                                                         CABIN
      PWI = EXT(LL)                                                       CABIN
      LL = LL + 1                                                         CABIN
      DWI = EXT(LL)                                                       CABIN
      PVPW = PV - PWI                                                     CABIN
      IF((PVPRME-PWI)/PVPW .LT. 0.) DWI = DWI*PVPW/(PV-PVPRME)            CABIN
      KK = KK + 1                                                         CABIN
      IF(EXT(KK)+DWI .LT. 0.0) DWI = -EXT(KK)                             CABIN
      EXT(KK) = EXT(KK) + DWI                                             CABIN
      SUMWL = SUMWL + DWI                                                 CABIN
      QL = DWI*POL(LAMDA,T(J))/TINC                                       CABIN
      Q(J) = Q(J) + QL                                                    CABIN
   60 CONTINUE                                                            CABIN
      RETURN                                                              CABIN
C                                                                         CABIN
      SUBROUTINE ERR(NUMB)                                                CABIN
      CALL TOPLIN                                                         CABIN
      WRITE(6,100) NUMB                                                   CABIN
  100 FORMAT(1H0 131(1H*)//1X'THE 'A3,' ARGUMENT IN THE CALL DOES NOT HA  CABIN
     XVE THE CORRECT NUMBER OF VALUES. EXECUTION TERMINATED IN SUBROUTIN  CABIN
     XE CABIN'//1X 131(1H*))                                              CABIN
      CALL WLKBCK                                                         CABIN
      CALL EXIT                                                           CABIN
C                                                                         CABIN
      SUBROUTINE QSUM                                                     CABIN
      IF(EXPLCT) DTAU(J) = DTAU(J) + HA                                   CABIN
      Q(J) = Q(J) + HA*(TC-T(J))                                          CABIN
      QL = HA*(TC-T(J))*TINC                                              CABIN
      SUMQL = SUMQL + QL                                                  CABIN
      XLAM = POL(LAMDA,T(J))                                              CABIN
      PWI = PO*EXP(XLAM/RV/(TO-TZ)*(T(J)-TO)/(T(J)-TZ)*CONV)              CABIN
      LL = LL + 1                                                         CABIN
      EXT(LL) = PWI                                                       CABIN
      DWI = HA/CPC/PC*(PV-PWI)*TINC                                       CABIN
      LL = LL + 1                                                         CABIN
      EXT(LL) = DWI                                                       CABIN
      SUMWL = SUMWL + DWI                                                 CABIN
      RETURN                                                              CABIN
      END                                                                 CABIN
```

```
      SUBROUTINE CMPRSS(SPR,IPR,A,B,L28,$)
C
C THIS SUBROUTINE REDUCES THE COEFFICIENT MATRIX FOR PFCS
C
C
      DIMENSION A(1), B(1), L28(1)
C
C
C
C LOCATE RELATIVE PRESSURE NODE NUMBER (NPR) OF ACTUAL PRESSURE
C NODE NUMBER (IPR) WHICH HAS A SPECIFIED PRESSURE (SPR)
C
      MPRN = L28(1)
      DO 5 NPR=1,MPRN
      IF(IPR .EQ. L28(NPR+1)) GO TO 10
    5 CONTINUE
      RETURN 6
C
C CALCULATE THE STARTING LOCATION OF COLUMN NPR
C
   10 LOC = (NPR-1)*NPR/2
      LD = LOC + NPR + NPR
C
C MODIFY THE RIGHT HAND SIDE
C
      DO 20 J=1,MPRN
      IF(J-NPR) 12,20,15
   12 B(J) = B(J) - SPR*A(LOC+J)
      GO TO 20
   15 B(J-1) = B(J) - SPR*A(LD)
      LD = LD + J
   20 CONTINUE
C
C CALCULATE THE STARTING LOCATION OF ROW NPR
C
      LD = LOC + NPR
      NPK = NPR
C
C DELETE COLUMN NPR
C
      NPR = NPR + 1
      DO 30 J=NPR,MPRN
      DO 25 L=1,J
C
C DELETE ROW NPR
C
      IF(L.EQ. NPK) GO TO 25
      LOC = LOC + 1
      A(LOC) = A(LD+L)
   25 CONTINUE
      LD = LD + J
   30 CONTINUE
C
C DELETE ACTUAL PRESSURE NODE IPR FROM THE LIST OF ACTUAL PRESSURE NODES
C
      DO 40 J=NPR,MPRN
      L28(J) = L28(J+1)
   40 CONTINUE
      L28(1) = L28(1) - 1
```

```
      RETURN                                                              CMPRSS
      END                                                                 CMPRSS
```

```
      SUBROUTINE CNFAST
C     AN EXPLICIT EXECUTION SUBROUTINE FOR SINDA   FORTRAN V
C     THE SHORT PSEUDO COMPUTE SEQUENCE IS REQUIRED
C     NODES WITH CSG BELOW DTIMEI RECEIVE STEADY STATE SOLUTION
C     NO BACKING UP IS DONE OR ALLOWED
      INCLUDE COMM,LIST
      INCLUDE DEFF,LIST
      IF(KON(5).LE.0) KON(5) = 1
      IF(CON(8).LE.0.) CON(8) = 1.E+8
      IF(CON(9).LE.0.) CON(9) = 1.0
      IF(CON(18).LE.0.) GO TO 999
      IF(CON(19).LE.0.) CON(19) = 1.E+8
      IF(CON(21).LE.0.) GO TO 998
      IF(KON(31).NE.0) GO TO 995
      PASS = -1.0
      NNC = NNA+NND
      IE = NTH
      NLA = NDIM
      NTH = NTH+NND
      NDIM = NDIM-NND
      IF(NDIM.LT.0) GO TO 997
      NN = NND+1
      TPRINT = CON(13)
      TSTEP = CON(21)
    5 TSUM = 0.0
      IF(CON(13)+CON(18).GT.CON(3)) CON(18) = CON(3)-CON(13)
   10 IF(TSTEP.GT.CON(8)) TSTEP = CON(8)
      IF(TSTEP.LT.CON(21)) TSTEP = CON(21)*1.000001
      IF(TSUM+TSTEP-CON(18)) 20,25,15
   15 TSTEP = CON(18)-TSUM
      GO TO 25
   20 IF(TSUM+2.0*TSTEP.GT.CON(18)) TSTEP = 0.5*(CON(18)-TSUM)
   25 CON(2) = TSTEP
      CON(1) = TPRINT+TSUM+TSTEP
      CON(14) = 0.5*(CON(1)+CON(13))
      DO 30 I = 1,NND
      Q(I) = 0.0
      LE = IE+I
   30 X(LE) = 0.0
      IF(NNA.LE.0) GO TO 40
      DO 35 I = NN,NNC
      Q(I) = 0.0
   35 CONTINUE
   40 KON(12) = 0
      CALL VARBL1
      IF(KON(12).NE.0) GO TO 10
      IF(PASS.GT.0.) GO TO 45
      PASS = 1.0
      CON(1) = TPRINT
      CON(2) = 0.0
      CALL OUTCAL
      CON(1) = TPRINT+TSTEP
      CON(2) = TSTEP
   45 J1 = 0
      J2 = 1
      DO 85 I = 1,NND
      LE = IE+I
      INCLUDE VARC,LIST
      INCLUDE VARQ,LIST
```

```
   50 J1 = J1+1                                                        CNFAST
      LG = FLD(5,16,NSQ1(J1))                                          CNFAST
      IF(LG.EQ.0) GO TO 85                                             CNFAST
      LTA = FLD(22,14,NSQ1(J1))                                        CNFAST
      INCLUDE VARG,LIST                                                CNFAST
      IF(FLD(3,1,NSQ1(J1)).EQ.0) GO TO 55                              CNFAST
      T1 = T(I)+460.0                                                  CNFAST
      T2 = T(LTA)+460.0                                                CNFAST
      GV = G(LG)*(T1*T1+T2*T2)*(T1+T2)                                 CNFAST
      GO TO 60                                                         CNFAST
   55 GV = G(LG)                                                       CNFAST
   60 QDOT = GV*(T(LTA)-T(I))                                  *NEW    CNFAST
      Q(I) = Q(I) + QDOT                                       *NEW    CNFAST
      X(LE) = X(LE)+GV                                         **-1    CNFAST
      IF(LTA.GT.NND.OR.FLD(21,1,NSQ1(J1)).EQ.1) GO TO 65               CNFAST
      LEA = IE+LTA                                                     CNFAST
      X(LEA) = X(LEA)+GV                                               CNFAST
      Q(LTA) = Q(LTA) - QDOT                                   *NEW    CNFAST
   65 IF(NSQ1(J1).GT.0) GO TO 50                               **-1    CNFAST
   85 CONTINUE                                                         CNFAST
      CKM = 1.E+8                                                      CNFAST
      TCGM = 0.0                                                       CNFAST
      DO 105 I = 1,NND                                                 CNFAST
      LE = IE+I                                                        CNFAST
      T1 = C(I)/X(LE)                                                  CNFAST
      IF(T1.GE.CKM) GO TO 90                                           CNFAST
      CKM = T1                                                         CNFAST
      KON(35) = I                                                      CNFAST
   90 IF(TSTEP.GT.T1) GO TO 95                                         CNFAST
      T1 = T(I) + TSTEP*Q(I)/C(I)                              *NEW    CNFAST
      GO TO 100                                                **-1    CNFAST
   95 T1 = T(I) + T1*Q(I)/C(I)                                 *NEW    CNFAST
  100 T2 = ABS(T1-T(I))                                        **-1    CNFAST
      T(I) = T1                                                        CNFAST
      IF(T2.LT.TCGM) GO TO 105                                         CNFAST
      TCGM = T2                                                        CNFAST
      KON(15) = I                                                      CNFAST
  105 CONTINUE                                                         CNFAST
      CON(15) = TCGM                                                   CNFAST
      CON(17) = CKM                                                    CNFAST
      IF(CKM.LE.0.) GO TO 996                                          CNFAST
      IF(NNA.LE.0) GO TO 160                                           CNFAST
      LAX = KON(5)                                                     CNFAST
      DAMPN = CON(9)                                                   CNFAST
      DAMPO = 1.0-DAMPN                                                CNFAST
      DO 150 I = 1,LAX                                                 CNFAST
      KON(20) = I                                                      CNFAST
      RLX = 0.0                                                        CNFAST
      JJ1 = J1                                                         CNFAST
      JJ2 = J2                                                         CNFAST
      DO 145 L = NN,NNC                                                CNFAST
      SUMC = 0.0                                                       CNFAST
      SUMCV = 0.0                                                      CNFAST
      IF(I.GT.1) GO TO 6000                                            CNFAST
      INCLUDE VRQ2,LIST                                                CNFAST
  110 JJ1 = JJ1+1                                                      CNFAST
      LG = FLD(5,16,NSQ1(JJ1))                                         CNFAST
      LTA = FLD(22,14,NSQ1(JJ1))                                       CNFAST
      IF(I.GT.1) GO TO 4000                                            CNFAST
      INCLUDE VRG2,LIST                                                CNFAST
C     CHECK FOR RADIATION CONDUCTOR                                    CNFAST
      IF(FLD(3,1,NSQ1(JJ1)).EQ.0) GO TO 115                            CNFAST
      T1 = T(L)+460.0                                                  CNFAST
```

```
      T2 = T(LTA)+460.0                                          CNFAST
      GV = G(LG)*(T1*T1+T2*T2)*(T1+T2)                           CNFAST
      GO TO 120                                                  CNFAST
  115 GV = G(LG)                                                 CNFAST
      T2 = T(LTA)                                                CNFAST
  120 SUMC = SUMC+GV                                             CNFAST
      SUMCV = SUMCV+GV*T2                                        CNFAST
C     CHECK FOR LAST CONDUCTOR TO THIS NODE                      CNFAST
      IF(NSQ1(JJ1).GT.0) GO TO 110                               CNFAST
      T1 = DAMPN*(SUMCV+Q(L))/SUMC+DAMPD*T(L)                    CNFAST
      T2 = ABS(T(L)-T1)                                          CNFAST
      IF(RLX.GE.T2) GO TO 140                                    CNFAST
      RLX = T2                                                   CNFAST
      KON(37) = L                                                CNFAST
  140 T(L) = T1                                                  CNFAST
  145 CONTINUE                                                   CNFAST
      IF(RLX.LE.CON(19)) GO TO 155                               CNFAST
  150 CONTINUE                                                   CNFAST
  155 CON(30) = RLX                                              CNFAST
  160 CALL VARBL2                                                CNFAST
      CON(13) = CON(1)                                           CNFAST
      TSUM = TSUM+TSTEP                                          CNFAST
      TSTEP = CKM                                                CNFAST
      IF(TSUM.LT.CON(18)) GO TO 10                               CNFAST
      TPRINT = TPRINT+TSUM                                       CNFAST
      CALL OUTCAL                                                CNFAST
      IF(CON(1)*1.000001.LT.CON(3)) GO TO 5                      CNFAST
      NTH = IE                                                   CNFAST
      NDIM = NLA                                                 CNFAST
      RETURN                                                     CNFAST
  995 WRITE(6,885)                                               CNFAST
      GO TO 1000                                                 CNFAST
  996 WRITE (6,886)                                              CNFAST
      GO TO 1000                                                 CNFAST
  997 WRITE(6,887) NDIM                                          CNFAST
      GO TO 1000                                                 CNFAST
  998 WRITE (6,888)                                              CNFAST
      GO TO 1000                                                 CNFAST
  999 WRITE(6,889)                                               CNFAST
 1000 CALL OUTCAL                                                CNFAST
      CALL EXIT                                                  CNFAST
  885 FORMAT(46H CNFAST REQUIRES SHORT PSEUDO-COMPUTE SEQUENCE)  CNFAST
  886 FORMAT(22H C/SK ZERO OR NEGATIVE)                          CNFAST
 887  FORMAT(I8,20H LOCATIONS AVAILABLE)                         CNFAST
  888 FORMAT(10H NO DTIMEL)                                      CNFAST
  889 FORMAT(19H NO OUTPUT INTERVAL)                             CNFAST
      END                                                        CNFAST
```

```
      SUBROUTINE COMBIN(NTAPE, KT, INC, IUNIT)                              COMBIN
C                                                                           COMBIN
      DIMENSION NBUFR(27), ALPHA(15), XSTART(7), XSTOP(7)            *NEW   COMBIN
C                                                                    *NEW   COMBIN
      COMMON /XYARY/ DATA(1)                                         *NEW   COMBIN
C                                                                    *NEW   COMBIN
      DATA ALPHA /1HA,1HB,1HC,1HD,1H5,1H6,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,  **-1   COMBIN
     1            1HL,1HM/                                                  COMBIN
C                                                                    *NEW   COMBIN
C                                                                    *NEW   COMBIN
      WRITE(6,3)                                                            COMBIN
    3 FORMAT(1H110X30HOUTPUT FROM COMBINE SUBROUTINE//)                     COMBIN
      IF(IUNIT .EQ. 0) IUNIT = 7                                            COMBIN
      REWIND KT                                                             COMBIN
      IF(NTAPE.GT.0) GO TO 7                                                COMBIN
      NTAPE = -NTAPE                                                        COMBIN
      READ(5,6) (XSTART(I),XSTOP(I),I=1,NTAPE)                              COMBIN
    6 FORMAT(14F5.3)                                                        COMBIN
    7 IUNIT1=IUNIT-1                                                        COMBIN
      DO 39 L=1,NTAPE                                                       COMBIN
      M=0                                                                   COMBIN
      N=0                                                                   COMBIN
      I=L+IUNIT1                                                            COMBIN
      REWIND I                                                              COMBIN
   15 READ(I) (NBUFR(J),J=1,26), NSL, (DATA(J),J=1,NSL)              *NEW   COMBIN
      NBUFR(27) = NSL                                                *NEW   COMBIN
      IF(L-1),15                                                     **-1   COMBIN
    9 READ(I)TIME,(DATA(K),K=1,NTOTAL)                                      COMBIN
      IF(TIME-XSTART(L))9                                                   COMBIN
      IF(TIME-XTIME),21,30                                                  COMBIN
      WRITE(6,12)                                                           COMBIN
   12 FORMAT(1H010X34HTAPES ARE NOT IN THE CORRECT ORDER)                   COMBIN
      CALL EXIT                                                             COMBIN
      WRITE(KT) NBUFR, (DATA(J),J=1,NSL)                             *NEW   COMBIN
      NTOTAL = 0                                                     **-1   COMBIN
      DO 18 J=17,27                                                         COMBIN
   18 NTOTAL=NTOTAL+NBUFR(J)                                                COMBIN
   21 READ(I)TIME,(DATA(K),K=1,NTOTAL)                                      COMBIN
      IF(TIME)24                                                            COMBIN
      IF(TIME-XSTART(L))21                                                  COMBIN
      IF(XSTOP(L))27,27                                                     COMBIN
      IF(TIME-XSTOP(L))27                                                   COMBIN
      TIME=-TIME                                                            COMBIN
   24 IF(L-NTAPE)33,30,33                                                   COMBIN
   27 M=M-1                                                                 COMBIN
      IF(M),,21                                                             COMBIN
      M=INC                                                                 COMBIN
      XTIME=TIME                                                            COMBIN
   30 WRITE(KT) TIME,(DATA(K),K=1,NTOTAL)                                   COMBIN
      IF(N.EQ.0) WRITE(6,31) TIME,L                                         COMBIN
   31 FORMAT(13X F10.5, 26H HAS BEEN LOADED FROM TAPE I2)                   COMBIN
      N=1                                                                   COMBIN
      IF(TIME),21,21                                                        COMBIN
      END FILE KT                                                           COMBIN
      REWIND KT                                                             COMBIN
   33 REWIND I                                                              COMBIN
      WRITE(6,36)L,XTIME                                                    COMBIN
   36 FORMAT(13X 4HTAPE I2, 10H ENDING AT F10.5,     28H HAS BEEN LOADED    COMBIN
     1 ON NEW TAPE/)                                                        COMBIN
```

```
      39 CONTINUE                                                     COMBIN
         IF(K.LE.15)WRITE(6,42)NTAPE,ALPHA(KT)                        COMBIN
      42 FORMAT(//10X9HDATA FROM I2,33H TAPES HAS BEEN COMBINED ON UNIT A2) COMBIN
         RETURN                                                       COMBIN
         END                                                          COMBIN
```

```
      SUBROUTINE CONDT1(NLOC)                                          CONDT1
C                                                                      CONDT1
      DIMENSION NLOC(1)                                                CONDT1
      COMMON /ARRAY / NDATA(1)                                         CONDT1
C                                                                      CONDT1
      COMMON /TEMP  / T  (1)                                           CONDT1
      COMMON /COND  / G  (1)                                           CONDT1
      COMMON /FIXCON/ CON(1)                                           CONDT1
      COMMON /DIMENS/ NND, NNA, NNT, NGT                               CONDT1
C                                                                      CONDT1
      IF(MOD(NLOC(1),4) .EQ. 0) GO TO 20                               CONDT1
      CALL TOPLIN                                                      CONDT1
      WRITE(6,10) NLOC(1)                                              CONDT1
   10 FORMAT(78HO* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONDT1 FOR CONDT1
     1 CONDUCTION DATA, IC = I5, 7H  * * *)                            CONDT1
      CALL WLKBCK                                                      CONDT1
      CALL EXIT                                                        CONDT1
C                                                                      CONDT1
   20 IC = NLOC(1)                                                     CONDT1
      DO 100 I=1,IC,4                                                  CONDT1
      NG    = NLOC(I+1)                                                CONDT1
      NLT   = NLOC(I+2)                                                CONDT1
      NTIME = NLOC(I+3)                                                CONDT1
      NTEMP = NLOC(I+4)                                                CONDT1
      IF(NG  .GT. NGT) GO TO 40                                        CONDT1
      IF(NLT .LE. NNT) GO TO 80                                        CONDT1
   40 CALL TOPLIN                                                      CONDT1
      WRITE(6,60) (NLOC(I+J),J=1,4)                                    CONDT1
   60 FORMAT(55HO* * *  ERROR IN CONDUCTION DATA INPUT TO CONDT1  * * * CONDT1
     1 // 5HONG = I4, 6H NLT = I5, 8H NTIME = I6, 8H NTEMP = I6)       CONDT1
      CALL WLKBCK                                                      CONDT1
      CALL EXIT                                                        CONDT1
C                                                                      CONDT1
   80 CALL DIDEG1(CON(1),NDATA(NTIME),FTIME)                           CONDT1
      CALL DIDEG1(T(NLT),NDATA(NTEMP),FTEMP)                           CONDT1
      G(NG) = FTIME*FTEMP                                              CONDT1
  100 CONTINUE                                                         CONDT1
C                                                                      CONDT1
      RETURN                                                           CONDT1
      END                                                              CONDT1
```

CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2 CONDT2

```
      SUBROUTINE CONDT2(NLOC)                                          CONDT2
C                                                                      CONDT2
      DIMENSION NLOC(1)                                                CONDT2
      COMMON /ARRAY / NDATA(1)                                         CONDT2
C                                                                      CONDT2
      COMMON /COND  / G  (1)                                           CONDT2
      COMMON /DIMENS/ NND, NNA, NNT, NGT                               CONDT2
C                                                                      CONDT2
      EQUIVALENCE (NTEMP,FTEMP)                                        CONDT2
C                                                                      CONDT2
C                                                                      CONDT2
      IF(MOD(NLOC(1),3) .EQ. 0) GO TO 20                               CONDT2
      CALL TOPLIN                                                      CONDT2
      WRITE(6,10) NLOC(1)                                              CONDT2
   10 FORMAT(78H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONDT2 FOR CONDT2
     1 CONDUCTION DATA, IC = I5, 7H  * * *)                            CONDT2
      CALL WLKBCK                                                      CONDT2
      CALL EXIT                                                        CONDT2
C                                                                      CONDT2
   20 IC = NLOC(1)                                                     CONDT2
      DO 100 I=1,IC,3                                                  CONDT2
      NG    = NLOC(I+1)                                                CONDT2
      NTIME = NLOC(I+2)                                                CONDT2
      NTEMP = NLOC(I+3)                                                CONDT2
      IF(NG .LE. NGT) GO TO 80                                         CONDT2
      CALL TOPLIN                                                      CONDT2
      WRITE(6,60) (NLOC(I+J),J=1,3)                                    CONDT2
   60 FORMAT(55H0* * *  ERROR IN CONDUCTION DATA INPUT TO CONDT2  * * * CONDT2
     1 // 5H0NG = I4, 8H NTIME = I6, 8H FTEMP = G13.8)                 CONDT2
      CALL WLKBCK                                                      CONDT2
      CALL EXIT                                                        CONDT2
C                                                                      CONDT2
   80 CALL DIDEG1(CON(1),NDATA(NTIME),FTIME)                           CONDT2
      G(NG) = FTIME*FTEMP                                              CONDT2
  100 CONTINUE                                                         CONDT2
C                                                                      CONDT2
      RETURN                                                           CONDT2
      END                                                              CONDT2
```

```
      SUBROUTINE CONV1(LLOC,MLOC,NLOC)
C
      LOGICAL LCP, LMU, LTC
C
      DIMENSION LLOC(1), MLOC(1), NLOC(1)
      DIMENSION RDATA(1)
C
      COMMON /ARRAY/ NDATA(1)
      COMMON /TEMP / T    (1)
      COMMON /COND / G    (1)
      COMMON /DIMENS/ NND, NNA, NNT, NGT
C
      EQUIVALENCE (RDATA,NDATA)
      EQUIVALENCE (NAHT,AHT), (NX,X), (NF1,F1), (NF2,F2)
C
      DATA MAX1 /65000/
C
C
      IF(LLOC(1) .EQ. 8) GO TO 20
      CALL TOPLIN
      WRITE(6,10) LLOC(1)
   10 FORMAT(71H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV1 FOR
     1FLOW DATA, IC =  I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
   20 L2 = LLOC(2)
      L7 = LLOC(7)
C
      IF(NDATA(L2) .GT. 0) GO TO 40
      CALL TOPLIN
      WRITE(6,30) NDATA(L2)
   30 FORMAT(72H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV1 FOR
     1FLOW RATES, IC = I5, 7H  * * *)
      GO TO 60
C
   40 IF(MOD(NDATA(L7),6) .EQ. 0) GO TO 75
      CALL TOPLIN
      WRITE(6,50) NDATA(L7)
   50 FORMAT(77H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV1 FOR
     1FLUID TYPE DATA, IC = I5, 7H  * * *)
   60 WRITE(6,70) (MLOC(I),I=2,7)
   70 FORMAT(5H0AW = I6, 6H APR = I6, 6H AGF = I6, 6H AVP = I6,
     1 7H AIFR = I6, 6H AFT = I6)
      CALL WLKBCK
      CALL EXIT
C
   75 IF(MLOC(1) .EQ. 5) GO TO 80
      CALL TOPLIN
      WRITE(6,76) MLOC(1)
   76 FORMAT(78H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV1 FOR
     1FLUID PROPERTIES, IC = I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
   80 NX = MLOC(2)
      IF(NX .LT. 1 .OR. NX .GT. MAX1) GO TO 82
      LCP = .TRUE.
      NCP = NX
```

```
      GO TO 84                                                                    CONV1
   82 LCP = .FALSE.                                                               CONV1
      CP  = X                                                                     CONV1
   84 NX  = NLOC(4)                                                               CONV1
      IF(NX .LT. 1 .OR. NX .GT. MAX1) GO TO 86                                    CONV1
      LMU = .TRUE.                                                                CONV1
      NMU = NX                                                                    CONV1
      GO TO 88                                                                    CONV1
   86 LMU = .FALSE.                                                               CONV1
      XMU = X                                                                     CONV1
   88 NX  = NLOC(5)                                                               CONV1
      IF(NX .LT. 1 .OR. NX .GT. MAX1) GO TO 92                                    CONV1
      LTC = .TRUE.                                                                CONV1
      NTC = NX                                                                    CONV1
      GO TO 94                                                                    CONV1
   92 LTC = .FALSE.                                                               CONV1
      TC  = X                                                                     CONV1
C                                                                                 CONV1
   94 IF(MOD(NLOC(1),8) .EQ. 0) GO TO 100                                         CONV1
      CALL TOPLIN                                                                 CONV1
      WRITE(6,95) NLOC(1)                                                         CONV1
   95 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV1 FOR           CONV1
     1CONVECTION DATA, NC = I5, 7H  * * *)                                        CONV1
      CALL WLKBCK                                                                 CONV1
      CALL EXIT                                                                   CONV1
C                                                                                 CONV1
  100 IC = NLOC(1)                                                                CONV1
      DO 220 I=1,IC,8                                                             CONV1
      NG    = NLOC(I+1)                                                           CONV1
      NAHT  = NLOC(I+2)                                                           CONV1
      ITUBE = NLOC(I+3)                                                           CONV1
      NFL   = NLOC(I+4)                                                           CONV1
      ITYPE = NLOC(I+5)                                                           CONV1
      NX    = NLOC(I+6)                                                           CONV1
      NF1   = NLOC(I+7)                                                           CONV1
      NF2   = NLOC(I+8)                                                           CONV1
      LTYPE = ITYPE*6 - 5                                                         CONV1
      IF(NG    .GT. NGT) GO TO 110                                                CONV1
      IF(NFL   .GT. NNT) GO TO 110                                                CONV1
      IF(ITUBE .GT. NDATA(L2)) GO TO 110                                          CONV1
      IF(LTYPE .LT. NDATA(L7)) GO TO 170                                          CONV1
  110 CALL TOPLIN                                                                 CONV1
      WRITE(6,120)                                                                CONV1
  120 FORMAT(54H0* * *  ERROR IN CONVECTION DATA INPUT TO CONV1  * * *)           CONV1
  150 WRITE(6,160) (NLOC(I+J),J=1,8)                                              CONV1
  160 FORMAT(5H0NG = I4, 6H AHT = G13.8, 8H ITUBE = I4, 6H NFL = I5,              CONV1
     1 8H ITYPE = I4, 4H X = G13.8, 5H F1 = G13.8, 5H F2 = G13.8)                 CONV1
      CALL WLKBCK                                                                 CONV1
      CALL EXIT                                                                   CONV1
C                                                                                 CONV1
  170 LTYPE = L7 + LTYPE                                                          CONV1
      WP  = RDATA(LTYPE)                                                          CONV1
      CSA = RDATA(LTYPE+1)                                                        CONV1
      IF(LCP) CALL D1DEG1(T(NFL),NDATA(NCP), CP)                                  CONV1
      IF(LMU) CALL D1DEG1(T(NFL),NDATA(NMU),XMU)                                  CONV1
      IF(LTC) CALL D1DEG1(T(NFL),NDATA(NTC), TC)                                  CONV1
      RE = 4.0*ABS(RDATA(L2+ITUBE))/XMU/WP                               *NEW     CONV1
      PR = XMU*CP/TC                                                     **-1     CONV1
      D  = 4.0*CSA/WP                                                             CONV1
      IF(RE .GT. 2000.0) GO TO 180                                                CONV1
      TEMP = X/D/RE/PR                                                            CONV1
      H = TC/D*(3.66*F1 +(0.0155*F2/(TEMP+0.015*CBRT(TEMP))))                     CONV1
      GO TO 200                                                                   CONV1
```

```
  180 IF(RE .LT. 6400.0) GO TO 190                                        CONV1
      H = 0.023*TC/D*RE**.8*CBRT(PR)                                      CONV1
      GO TO 200                                                           CONV1
  190 H = 0.116*TC/D*(CBRT(RE*RE)-125.0)*CBRT(PR)                         CONV1
  200 G(NG) = H*AHT                                                       CONV1
  220 CONTINUE                                                            CONV1
C                                                                         CONV1
      RETURN                                                              CONV1
      END                                                                 CONV1
```

```
      SUBROUTINE CONV2(LLOC,MLOC,NLOC)
C
      LOGICAL LCP, LMU, LTC
C
      DIMENSION LLOC(1), MLOC(1), NLOC(1)
      DIMENSION RDATA(1)
C
      COMMON /ARRAY/ NDATA(1)
      COMMON /TEMP / T    (1)
      COMMON /COND / G    (1)
      COMMON /DIMENS/ NND, NNA, NNT, NGT
C
      EQUIVALENCE (RDATA,NDATA)
      EQUIVALENCE (NAMT,AMT), (NX,X)
C
      DATA MAXI /65000/
C
C
      IF(LLOC(1) .EQ. 8) GO TO 20
      CALL TOPLIN
      WRITE(6,10) LLOC(1)
   10 FORMAT(71H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR
     1FLOW DATA, IC =  I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
   20 L2 = LLOC(2)
      L7 = LLOC(7)
C
      IF(NDATA(L2) .GT. 0) GO TO 40
      CALL TOPLIN
      WRITE(6,30) NDATA(L2)
   30 FORMAT(72H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR
     1FLOW RATES, IC = I5, 7H  * * *)
      GO TO 60
C
   40 IF(MOD(NDATA(L7),6) .EQ. 0) GO TO 75
      CALL TOPLIN
      WRITE(6,50) NDATA(L7)
   50 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR
     1FLUID TYPE DATA, IC = I5, 7H  * * *)
   60 WRITE(6,70) (MLOC(I),I=2,7)
   70 FORMAT(5H0AW = I6, 6H APR = I6, 6H AGF = I6, 6H AVP = I6,
     1 7H AIFR = I6, 6H AFT = I6)
      CALL WLKBCK
      CALL EXIT
C
   75 IF(MLOC(1) .EQ. 5) GO TO 80
      CALL TOPLIN
      WRITE(6,76) MLOC(1)
   76 FORMAT(78H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR
     1FLUID PROPERTIES, IC = I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
   80 NX = MLOC(2)
      IF(NX .LT. 1 .OR. NX .GT. MAXI) GO TO 82
      LCP = .TRUE.
      NCP = NX
```

```
      GO TO 84                                                          CONV2
   82 LCP = .FALSE.                                                     CONV2
      CP  = X                                                           CONV2
   84 NX  = NLOC(4)                                                     CONV2
      IF(NX .LT. 1 .OR. NX .GT. MAXI) GO TO 86                          CONV2
      LMU = .TRUE.                                                      CONV2
      NMU = NX                                                          CONV2
      GO TO 88                                                          CONV2
   86 LMU = .FALSE.                                                     CONV2
      XMU = X                                                           CONV2
   88 NX  = NLOC(5)                                                     CONV2
      IF(NX .LT. 1 .OR. NX .GT. MAXI) GO TO 92                          CONV2
      LTC = .TRUE.                                                      CONV2
      NTC = NX                                                          CONV2
      GO TO 94                                                          CONV2
   92 LTC = .FALSE.                                                     CONV2
      TC  = X                                                           CONV2
C                                                                       CONV2
   94 IF(MOD(NLOC(1),6) .EQ. 0) GO TO 100                               CONV2
      CALL TOPLIN                                                       CONV2
      WRITE(6,95) NLOC(1)                                               CONV2
   95 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV2 FOR CONV2
     1CONVECTION DATA, IC = I5, 7H  * * *)                              CONV2
      CALL WLKBCK                                                       CONV2
      CALL EXIT                                                         CONV2
C                                                                       CONV2
  100 IC = NLOC(1)                                                      CONV2
      DO 220 I=1,IC,6                                                   CONV2
      NG    = NLOC(I+1)                                                 CONV2
      NAHT  = NLOC(I+2)                                                 CONV2
      ITUBE = NLOC(I+3)                                                 CONV2
      NFL   = NLOC(I+4)                                                 CONV2
      ITYPE = NLOC(I+5)                                                 CONV2
      NHST  = NLOC(I+6)                                                 CONV2
      LTYPE = ITYPE*6 - 5                                               CONV2
      IF(NG    .GT. NGT) GO TO 110                                      CONV2
      IF(NFL   .GT. NNT) GO TO 110                                      CONV2
      IF(ITUBE .GT. NDATA(L2)) GO TO 110                                CONV2
      IF(LTYPE .LT. NDATA(L7)) GO TO 170                                CONV2
  110 CALL TOPLIN                                                       CONV2
      WRITE(6,120)                                                      CONV2
  120 FORMAT(54H0* * *  ERROR IN CONVECTION DATA INPUT TO CONV2  * * *) CONV2
  150 WRITE(6,160) (NLOC(I+J),J=1,6)                                    CONV2
  160 FORMAT(5H0NG = I4, 6H AHT = G13.8, 8H ITUBE = I4, 6H NFL = I5,    CONV2
     1 8H ITYPE = I4, 7H AHST = I6)                                     CONV2
      CALL WLKBCK                                                       CONV2
      CALL EXIT                                                         CONV2
C                                                                       CONV2
  170 LTYPE = LT + LTYPE                                                CONV2
      WP  = RDATA(LTYPE)                                                CONV2
      CSA = RDATA(LTYPE+1)                                              CONV2
      IF(LCP) CALL D1DEG1(T(NFL),NDATA(NCP), CP)                        CONV2
      IF(LMU) CALL D1DEG1(T(NFL),NDATA(NMU),XMU)                        CONV2
      IF(LTC) CALL D1DEG1(T(NFL),NDATA(NTC), TC)                        CONV2
      RE = 4.0*ABS(RDATA(L2+ITUBE))/XMU/WP                        *NEW  CONV2
      PR = XMU*CP/TC                                              **-1  CONV2
      D  = 4.0*CSA/WP                                                   CONV2
      CALL D1DEG1(RE,NDATA(NHST),ST)                                    CONV2
      H = TC/D*RE*CBRT(PR)*ST                                           CONV2
  200 G(NG) = H*AHT                                                     CONV2
  220 CONTINUE                                                          CONV2
C                                                                       CONV2
      RETURN                                                            CONV2
```

CONV2

END

```
      SUBROUTINE CONV3(MLOC,NLOC)
C
      DIMENSION MLOC(1),NLOC(1)
      DIMENSION RDATA(1)
C
      COMMON /ARRAY/ NDATA(1)
      COMMON /TEMP / T    (1)
      COMMON /COND / G    (1)
      COMMON /DIMENS/ NND, NNA, NNT, NGT
C
      EQUIVALENCE (RDATA,NDATA)
      EQUIVALENCE (NAHT,AHT)
C
      DATA MAXI /65000/
C
C
      IF(MLOC(1) .EQ. 6) GO TO 20
      CALL TOPLIN
      IF(LLOC(1) .EQ. 8) GO TO 20                                     *NEW
   10 FORMAT(71H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV3 FOR **-1
     1FLOW DATA, IC =  I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
   20 L2 = MLOC(2)
      L7 = MLOC(7)
C
      IF(NDATA(L2) .GT. 0) GO TO 80
      CALL TOPLIN
      WRITE(6,30) NDATA(L2)
   30 FORMAT(72H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV3 FOR
     1FLOW RATES, IC = I5, 7H  * * *)
   60 WRITE(6,70) (MLOC(I),I=2,7)
   70 FORMAT(5H0AW = I6, 6H APR = I6, 6H AGF = I6, 6H AVP = I6,
     1 7H AIFR = I6, 6H AFT = I6)
      CALL WLKBCK
      CALL EXIT
C
C
   80 IF(MOD(NLOC(1),4) .EQ. 0) GO TO 100
      CALL TOPLIN
      WRITE(6,90) NLOC(1)
   90 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO CONV3 FOR
     1CONVECTION DATA, IC = I5, 7H  * * *)
      CALL WLKBCK
      CALL EXIT
C
  100 IC = NLOC(1)
      DO 220 I=1,IC,4
      NG    = NLOC(I+1)
      NAHT  = NLOC(I+2)
      ITUBE = NLOC(I+3)
      NHW   = NLOC(I+4)
      IF(NG     .GT. NGT) GO TO 110
      IF(ITUBE .LE. NDATA(L2)) GO TO 170
  110 CALL TOPLIN
      WRITE(6,120) (NLOC(I+J),J=1,4)
  120 FORMAT(54H0* * *  ERROR IN CONVECTION DATA INPUT TO CONV3  * * *
     1     // 5H0NG = I4, 6H AHT = G13.8, 8H ITUBE = I4, 6H AHW = I5)
```

```
      CALL WLKBCK                                                       CONV3
      CALL EXIT                                                         CONV3
C                                                                       CONV3
  170 CALL DIDEGI(RDATA(L2+ITUBE),NDATA(NHW),M)                         CONV3
  200 B(NG) = M*AHT                                                     CONV3
  220 CONTINUE                                                          CONV3
C                                                                       CONV3
      RETURN                                                            CONV3
      END                                                               CONV3
```

```
      SUBROUTINE FLOBAL(NPRN,L14,WIN,NPI,NPO,FROF,DWMX)
C                                                              *NEW     FLOBAL
      LOGICAL LVP, LIFR, LAR, LDP, COP                         **-1     FLOBAL
      LOGICAL LPR                                                       FLOBAL
C                                                              *NEW     FLOBAL
      DIMENSION RDATA(1), EXT(1)                                        FLOBAL
C                                                                       FLOBAL
      COMMON /ARRAY / NDATA(1)                                          FLOBAL
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9                    FLOBAL
      COMMON /FDATA / LVP, LIFR, LAR, LDP                               FLOBAL
      COMMON /FDATA / COP, LAO, NAO, AO, LMU, NMU, XMU, GC2             FLOBAL
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)                                FLOBAL
C                                                              **-1     FLOBAL
      EQUIVALENCE (RDATA,NDATA), (EXT,NEXT)                             FLOBAL
C                                                                       FLOBAL
C                                                              **-2     FLOBAL
      L20=NDATA(L14)-3                                                  FLOBAL
      L22 = NDATA(L14+2)                                                FLOBAL
      L23 = NDATA(L14+3)                                                FLOBAL
      L25 = NTH + 1                                                     FLOBAL
C                                                                       FLOBAL
C                                                                       FLOBAL
      IF(L23 .LT. 1) GO TO 602                                          FLOBAL
      L40 = NDATA(L23)                                                  FLOBAL
      IF(L40 .LT. 1) GO TO 602                                          FLOBAL
C                                                                       FLOBAL
      IF(LVP) GO TO 530                                                 FLOBAL
  510 WRITE(6,520) NDATA(L5), NDATA(L14+1)                              FLOBAL
  520 FORMAT(78H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO FLOBAL FOR FLOBAL
     1 VALVE POSITIONS, IC = I5, 7H  * * * // 8X 12HFOR NETWORK  A6)    FLOBAL
      CALL WLKBCK                                                       FLOBAL
      CALL EXIT                                                         FLOBAL
C                                                                       FLOBAL
  530 DO 600 J=1,L40                                                    FLOBAL
      L41=NDATA(L23+J)                                                  FLOBAL
      NV   = NDATA(L41+1)                                               FLOBAL
      NTS1 = NDATA(L41+2)                                               FLOBAL
      NTS2 = NDATA(L41+3)                                               FLOBAL
      E    = RDATA(L41+7)                                               FLOBAL
      IF(NV .GT. NDATA(L5)) GO TO 510                                   FLOBAL
      XS1  = RDATA(L5+NV)                                               FLOBAL
      IF(.NOT. COP) GO TO 540                                           FLOBAL
      RVS1 = 0.0                                                        FLOBAL
      RVS2 = 0.0                                                        FLOBAL
  540 IF(NTS1 .LT. 1) GO TO 560                                         FLOBAL
      IF(NTS1 .GT. NDATA(L2)) GO TO 590                                 FLOBAL
      RVS1 = RDATA(L2+NTS1)*E/XS1/XS1                                   FLOBAL
      RDATA(L4+NTS1) = 1.0/(1.0/RDATA(L4+NTS1)+RVS1)                    FLOBAL
  560 IF(NTS2 .LT. 1) GO TO 570                                         FLOBAL
      IF(NTS2 .GT. NDATA(L2)) GO TO 590                                 FLOBAL
      XS2  = 1.0 - XS1                                                  FLOBAL
      RVS2 = RDATA(L2+NTS2)*E/XS2/XS2                                   FLOBAL
      RDATA(L4+NTS2) = 1.0/(1.0/RDATA(L4+NTS2)+RVS2)                    FLOBAL
C                                                                       FLOBAL
  570 IF(.NOT. COP) GO TO 600                                           FLOBAL
      CALL LINECK(3)                                                    FLOBAL
      WRITE(6,580) NV, E, XS1, NTS1, RVS1, NTS2, RVS2                   FLOBAL
  580 FORMAT(/             7X 7HNV     = I10   , 8X 7HE      = G13.8,   FLOBAL
     1  5X 7HXS1    = G13.8, 5X 7HNTS1   = I10   , 8X 7HRVS1   = G13.8/ FLOBAL
```

```
     2  7X 7HNTS2  = I10  , 8X 7HRVS2  = G13.8 )                      FLOBAL
      GO TO 600                                                        FLOBAL
C                                                                      FLOBAL
  590 WRITE(6,595) NDATA(L14+1), NV, NTS1, NTS2                        FLOBAL
  595 FORMAT(57H0* * *  ERROR IN VALVE DATA INPUT TO FLOBAL FOR NETWORK FLOBAL
     1 A6, 7H  * * * // 8X 4HNV = I5, 7H NTS1 = I5, 7H NTS2 = I5)      FLOBAL
      CALL WLKBCK                                                      FLOBAL
      CALL EXIT                                                        FLOBAL
C                                                                      FLOBAL
C                                                                      FLOBAL
  600 CONTINUE                                                         FLOBAL
C                                                                      FLOBAL
C                                                                      FLOBAL
  602 M = NPRN*(NPRN+1)/2                                              FLOBAL
      NTH = NTH + NPRN + 1                                             FLOBAL
      L26 = NTH + 1                                                    FLOBAL
      NTH = NTH + M + 3                                                FLOBAL
      L27 = NTH + 1                                                    FLOBAL
      NTH = NTH + NPRN + 1                                             FLOBAL
      L28 = NTH + 1                                                    FLOBAL
      NTH = NTH + NPRN + 1                                             FLOBAL
      IF(NDIM .GE. NTH-L25+1) GO TO 610                                FLOBAL
C                                                                      FLOBAL
      NEED = NTH - NDIM -L25 + 1                                       FLOBAL
      CALL TOPLIN                                                      FLOBAL
      WRITE(6,605) NEED, NDATA(L14+1)                                  FLOBAL
  605 FORMAT(83H0* * *  INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR FLOW FLOBAL
     1BALANCING SUBROUTINE  * * * // 8X 5HSHORT I5, 23H LOCATIONS FOR NE FLOBAL
     2TWORK A6)                                                        FLOBAL
      CALL WLKBCK                                                      FLOBAL
      CALL EXIT                                                        FLOBAL
C                                                                      FLOBAL
  610 DO 620 J=L26,NTH                                                 FLOBAL
      NEXT(J) = 0                                                      FLOBAL
  620 CONTINUE                                                         FLOBAL
C                                                                      FLOBAL
      NEXT(L26) = M + 2                                                FLOBAL
      NEXT(L26+1) = NPRN                                               FLOBAL
      NEXT(L26+2) = NPRN                                               FLOBAL
      NEXT(L27)   = NPRN                                               FLOBAL
C                                                                      FLOBAL
      NEXT(L28) = NPRN                                                 FLOBAL
      DO 622 J=1,NPRN                                                  FLOBAL
      NEXT(L28+J) = NEXT(L25+J)                                        FLOBAL
  622 CONTINUE                                                         FLOBAL
C                                                                      FLOBAL
C ASSEMBLE COEFFICIENT MATRIX                                          FLOBAL
C                                                                      FLOBAL
      DO 625 J=4,L20,5                                                 FLOBAL
      K = L19 + J                                                      FLOBAL
      NTB = NDATA(K)                                                   FLOBAL
      NFRM = NDATA(K+1)                                                FLOBAL
      NTO  = NDATA(K+2)                                                FLOBAL
      NR   = MIN0 (NFRM,NTO)                                           FLOBAL
      NC   = MAX0 (NFRM,NTO)                                           FLOBAL
      NRNR = (NR+1)*NR/2 + 2                                           FLOBAL
      NRNC = (NC-1)*NC/2 + NR + 2                                      FLOBAL
      NCNC = (NC+1)*NC/2 + 2                                           FLOBAL
      GF   = RDATA(L4+NTB)                                             FLOBAL
      EXT(L26+NRNR) = EXT(L26+NRNR) + GF                               FLOBAL
      EXT(L26+NRNC) = EXT(L26+NRNC) - GF                               FLOBAL
      EXT(L26+NCNC) = EXT(L26+NCNC) + GF                               FLOBAL
  625 CONTINUE                                                         FLOBAL
```

```
      IF(.NOT. COP) GO TO 630                                        *NEW     FLOBAL
      CALL LINECK(2)                                                 *NEW     FLOBAL
      CALL GENOUT(1,1,0,'OMATRIX BEFORE REDUCTION')                           FLOBAL
      CALL LINECK(2)                                                 *NEW     FLOBAL
      CALL GENOUT(EXT(L28+1),1,NEXT(L28)  ,'OPRESSURE NODES')        *NEW     FLOBAL
      CALL LINECK(2)                                                 *NEW     FLOBAL
      CALL GENOUT(EXT(L26+3),1,NEXT(L26)-2,'OCOEFFECIENT MATRIX')    *NEW     FLOBAL
      CALL LINECK(2)                                                 *NEW     FLOBAL
      CALL GENOUT(EXT(L27+1),1,NEXT(L28)  ,'ORIGHT HAND SIDE')       *NEW     FLOBAL
C                                                                    **-3     FLOBAL
C IMPOSED FLOW RATES INTO RHS                                                 FLOBAL
C                                                                             FLOBAL
  630 IF(.NOT. LIFR) GO TO 690                                       *NEW     FLOBAL
      DO 680 J=1,NPRN                                                **-1     FLOBAL
      NPR = NEXT(L25+J)                                                       FLOBAL
      EXT(L27+J) = RDATA(L6+NPR)                                              FLOBAL
  680 CONTINUE                                                                FLOBAL
C                                                                             FLOBAL
C INLET FLOW RATE INTO RHS                                                    FLOBAL
C                                                                             FLOBAL
  690 N = NPRN                                                                FLOBAL
  700 IF(NPI .LT. 1) GO TO 780                                                FLOBAL
      NIFNR = NPI                                                             FLOBAL
      CALL PRN(NEXT(L25),N,NIFNR)                                             FLOBAL
      IF(N .LE. NPRN) GO TO 760                                               FLOBAL
C                                                                             FLOBAL
      NIFN = NPI                                                              FLOBAL
  720 CALL TOPLIN                                                             FLOBAL
      WRITE(6,740) NIFN, NDATA(L14+1)                                         FLOBAL
  740 FORMAT(69H0* * *  ERROR IN LOCATING PRESSURE NODE WITH IMPOSED FLO      FLOBAL
     1W RATE  * * * // 8X 4HNODE I5, 27H WAS NOT FOUND FOR NETWORK A6)        FLOBAL
      CALL BLKBCK                                                             FLOBAL
      CALL EXIT                                                               FLOBAL
C                                                                             FLOBAL
  760 EXT(L27+NIFNR) = WIN                                                    FLOBAL
C                                                                             FLOBAL
C SPECIFIED PRESSURES INTO COEFFICIENT MATRIX AND RHS                         FLOBAL
C                                                                             FLOBAL
  780 IF(L22 .LT. 1) GO TO 840                                                FLOBAL
      IF(NDATA(L22) .LT. 1) GO TO 840                                         FLOBAL
      L60 = NDATA(L22)                                                        FLOBAL
      DO 820 J=1,L60                                                          FLOBAL
      NSPRN  = NDATA(L22+J)                                                   FLOBAL
      CALL CMPRSS(RDATA(L3+NSPRN),NSPRN ,EXT(L26+3),EXT(L27+1),               FLOBAL
     1 NEXT(L28),$860)                                                        FLOBAL
  820 CONTINUE                                                                FLOBAL
      LPR = .FALSE.                                                  *NEW     FLOBAL
      RPR = 0.                                                       *NEW     FLOBAL
      IF(NPO) 920,920,845                                            *NEW     FLOBAL
C                                                                    *NEW     FLOBAL
  840 IF(NPO .LT. 1) GO TO 920                                       *NEW     FLOBAL
      LPR = .TRUE.                                                   *NEW     FLOBAL
      RPR = RDATA(L3+NPO)                                            *NEW     FLOBAL
      RDATA(L3+NPO) = 0.0                                            *NEW     FLOBAL
  845 CALL CMPRSS(RDATA(L3+NPO),NPO,EXT(L26+3),EXT(L27+1),           *NEW     FLOBAL
     X             NEXT(L28),$850)                                   *NEW     FLOBAL
      GO TO 920                                                      **-4     FLOBAL
C                                                                             FLOBAL
  850 NSPRN = NPO                                                             FLOBAL
  860 CALL TOPLIN                                                             FLOBAL
      WRITE(6,880) NSPRN, NDATA(L14+1)                                        FLOBAL
  880 FORMAT(70H0* * *  ERROR IN LOCATING PRESSURE NODE WITH PRESSURE SP      FLOBAL
     1ECIFIED  * * * // 8X 4HNODE I5, 27H WAS NOT FOUND FOR NETWORK A6)       FLOBAL
```

```
      CALL WLKBCK                                               ** FLOBAL
      CALL EXIT                                                 ** FLOBAL
C                                                               ** FLOBAL
C OUTLET PRESSURE INTO COEFFICIENT MATRIX AND RHS                   FLOBAL
C                                                                   FLOBAL
C                                                                   FLOBAL
C SOLVE FOR PRESSURES                                               FLOBAL
C                                                                   FLOBAL
  920 MPRN = NEXT(L28)                                              FLOBAL
      NEXT(L26) = MPRN*(MPRN+1)/2 + 2                               FLOBAL
      NEXT(L26+1) = MPRN                                            FLOBAL
      NEXT(L26+2) = MPRN                                            FLOBAL
      IF(.NOT. COP) GO TO 930                                       FLOBAL
      CALL LINECK(2)                                         *NEW   FLOBAL
      CALL GENOUT(1,1,0,'0MATRIX AFTER  REDUCTION')                 FLOBAL
      CALL LINECK(2)                                         *NEW   FLOBAL
      CALL GENOUT(EXT(L28+1),1,NEXT(L28)  ,'0PRESSURE NODES')*NEW   FLOBAL
      CALL LINECK(2)                                         *NEW   FLOBAL
      CALL GENOUT(EXT(L26+3),1,NEXT(L26)-2,'0COEFFECIENT MATRIX')  *NEW   FLOBAL
      CALL LINECK(2)                                         *NEW   FLOBAL
      CALL GENOUT(EXT(L27+1),1,NEXT(L28)  ,'0RIGHT HAND SIDE') *NEW FLOBAL
      CALL LINECK(2)                                         *NEW   FLOBAL
  930 CALL SYMSOL(EXT(L26+3),MPRN,EXT(L27+1),$1020)          **-3   FLOBAL
      IF(COP) CALL GENOUT(EXT(L27+1),1,NEXT(L28),'0PRESSURES') *NEW FLOBAL
      GO TO 1060                                             **-1   FLOBAL
C                                                                   FLOBAL
 1020 CALL TOPLIN                                                   FLOBAL
      WRITE(6,1040) NDATA(L14+1)                                    FLOBAL
 1040 FORMAT(61H0* * *  ERROR IN SOLVING PRESSURE/FLOW EQUATIONS FOR NET   FLOBAL
     1WORK A6, 7H  * * *)                                           FLOBAL
      CALL WLKBCK                                                   FLOBAL
      NTH = L25 - 1                                          *NEW   FLOBAL
      CALL OUTCAL                                                   FLOBAL
      CALL EXIT                                                     FLOBAL
C                                                                   FLOBAL
C                                                                   FLOBAL
C UPDATE PRESSURES                                                  FLOBAL
C                                                                   FLOBAL
 1060 DO 1080 J=1,MPRN                                              FLOBAL
      NPR = NEXT(L28+J)                                             FLOBAL
      RDATA(L3+NPR) = EXT(L27+J)                                    FLOBAL
 1080 CONTINUE                                                      FLOBAL
C                                                                   FLOBAL
C CALCULATE NEW FLOW RATES                                          FLOBAL
C                                                                   FLOBAL
      DO 1120 J=4,L20,5                                             FLOBAL
      K    = L14 + J                                                FLOBAL
      NTB  = NDATA(K)                                               FLOBAL
      NFRM = NDATA(K+1)                                             FLOBAL
      NTO  = NDATA(K+2)                                             FLOBAL
      NFRM = NEXT(L25+NFRM)                                         FLOBAL
      NTO  = NEXT(L25+NTO )                                         FLOBAL
      TEMP = RDATA(L4+NTB)*(RDATA(L3+NFRM)-RDATA(L3+NTO))           FLOBAL
      WNTB = RDATA(L2+NTB)                                          FLOBAL
      TEMP = WNTB + FRCF*(TEMP-WNTB)                                FLOBAL
      RDATA(L2+NTB) = TEMP                                          FLOBAL
      DWMX = AMAX1(ABS(TEMP/WNTB-1.0),DWMX)                         FLOBAL
C                                                                   FLOBAL
      IF(.NOT. COP) GO TO 1120                                      FLOBAL
      CALL LINECK(3)                                                FLOBAL
      PNFRM = RDATA(L3+NFRM) + APR                           *NEW   FLOBAL
      PNTO  = RDATA(L3+NTO ) + APR                           *NEW   FLOBAL
      WRITE(6,1100) NTB,NFRM,NTO,RDATA(L4+NTB),PNFRM,PNTO,WNTB,TEMP *NEW FLOBAL
```

```
      1100 FORMAT(/            7X 7HNTB   = I10  , 8X 7HNFRM  = I10 ,               **-2      FLOBAL
          1  8X 7HNTO    = I10  , 8X 7HGF     = G13.8/32X 7HP(NFRM) G13.8,          *NEW      FLOBAL
          2  5X 7HP(NTO)= G13.8, 5X 7HWOLD  = G13.8, 5X 7HWNEW  = G13.8)            **-1      FLOBAL
      1120 CONTINUE                                                                           FLOBAL
           IF( .NOT. LPR) GO TO 1160                                                *NEW      FLOBAL
           DO 1140  J=1,MPRN                                                        *NEW      FLOBAL
           NPR = NEXT(L28+J)                                                        *NEW      FLOBAL
           RDATA(L3+NPR) = RDATA(L3+NPR) + RPR                                      *NEW      FLOBAL
      1140 CONTINUE                                                                 *NEW      FLOBAL
           RDATA(L3+NPO) = RPR                                                      *NEW      FLOBAL
      1160 NTH = L25 - 1                                                            *NEW      FLOBAL
           RETURN                                                                   **-1      FLOBAL
           END                                                                                FLOBAL
```

```
      SUBROUTINE FLOCN1(LLOC,MLOC,NLOC)                                 FLOCN1
C                                                                       FLOCN1
      LOGICAL LCP                                                       FLOCN1
C                                                                       FLOCN1
      DIMENSION LLOC(1), MLOC(1), NLOC(1)                               FLOCN1
C                                                                       FLOCN1
      COMMON /TEMP  / T(1)                                              FLOCN1
      COMMON /COND  / G(1)                                              FLOCN1
      COMMON /DIMENS/ NND, NNA, NNT, NGT                                FLOCN1
C                                                                       FLOCN1
      EQUIVALENCE (NW,W)                                                FLOCN1
C                                                                       FLOCN1
C                                                                       FLOCN1
      IF(LLOC(1) .GT. 0) GO TO 15                                       FLOCN1
      CALL TOPLIN                                                       FLOCN1
      WRITE(6,10) LLOC(1)                                               FLOCN1
   10 FORMAT(73H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCN1 FOR FLOCN1
     1 FLOW RATES, IC = I5, 7H  * * *)                                  FLOCN1
      CALL WLKBCK                                                       FLOCN1
      CALL EXIT                                                         FLOCN1
C                                                                       FLOCN1
   15 NW = MLOC(1)                                                      FLOCN1
      IF(NW .LT. 1 .OR. NW .GT. 65000) GO TO 20                         FLOCN1
      LCP = .TRUE.                                                      FLOCN1
      CP  = W                                                           FLOCN1
      GO TO 25                                                          FLOCN1
   20 LCP = .FALSE.                                                     FLOCN1
      RCP = NW                                                          FLOCN1
C                                                                       FLOCN1
   25 IF(MOD(NLOC(1),3) .EQ. 0) GO TO 40                                FLOCN1
      CALL TOPLIN                                                       FLOCN1
      WRITE(6,30) NLOC(1)                                               FLOCN1
   30 FORMAT(83H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCN1 FOR FLOCN1
     1 FLOW CONDUCTION DATA, IC = I5, 7H  * * *)                        FLOCN1
      CALL WLKBCK                                                       FLOCN1
      CALL EXIT                                                         FLOCN1
C                                                                       FLOCN1
   40 NTB = LLOC(1)                                                     FLOCN1
      IC  = NLOC(1)                                                     FLOCN1
      DO 200 I=1,IC,3                                                   FLOCN1
      NG    = NLOC(I+1)                                                 FLOCN1
      NDL   = NLOC(I+2)                                                 FLOCN1
      ITUBE = NLOC(I+3)                                                 FLOCN1
      IF(NG     .GT. NGT) GO TO 60                                      FLOCN1
      IF(ITUBE .GT. NTB) GO TO 60                                       FLOCN1
      IF(NDL    .LE. NNT) GO TO 100                                     FLOCN1
   60 CALL TOPLIN                                                       FLOCN1
      WRITE(6,80) (NLOC(I+J),J=1,3)                                     FLOCN1
   80 FORMAT(60H0* * *  ERROR IN FLOW CONDUCTION DATA INPUT TO FLOCN1  * FLOCN1
     1 * * // 5H0NG = I4, 6H NDL = I5, 8H ITUBE = I4)                   FLOCN1
      CALL WLKBCK                                                       FLOCN1
      CALL EXIT                                                         FLOCN1
C                                                                       FLOCN1
  100 NW = LLOC(ITUBE+1)                                                FLOCN1
      IF(LCP) CALL DIDEG1(T(NDL),MLOC,CP)                               FLOCN1
      G(NG) = W*CP                                                      FLOCN1
  200 CONTINUE                                                          FLOCN1
C                                                                       FLOCN1
      RETURN                                                            FLOCN1
```

END FLOCN1

```
      SUBROUTINE FLOCN2(MLOC,CP,NLOC)                      *NEW    FLOCN2
C                                                          **-1    FLOCN2
      DIMENSION MLOC(1), NLOC(1)                                   FLOCN2
C                                                                  FLOCN2
      COMMON /COND  / G(1)                                         FLOCN2
      COMMON /DIMENS/ NND, NNA, NNT, NGT                           FLOCN2
C                                                                  FLOCN2
      EQUIVALENCE (NW,W)                                   *NEW    FLOCN2
C                                                          **-1    FLOCN2
C                                                                  FLOCN2
      IF(MLOC(1) .GT. 0) GO TO 20                                  FLOCN2
      CALL TOPLIN                                                  FLOCN2
      WRITE(6,10) MLOC(1)                                          FLOCN2
   10 FORMAT(73H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCN2 FOR   FLOCN2
     1 FLOW RATES, IC = I5, 7H  * * *)                             FLOCN2
      CALL WLKBCK                                                  FLOCN2
      CALL EXIT                                                    FLOCN2
C                                                                  FLOCN2
   20 IF(MOD(NLOC(1),2) .EQ. 0) GO TO 40                   *NEW    FLOCN2
      CALL TOPLIN                                          **-1    FLOCN2
      WRITE(6,30) NLOC(1)                                          FLOCN2
   30 FORMAT(83H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO FLOCN2 FOR   FLOCN2
     1 FLOW CONDUCTION DATA, IC = I5, 7H  * * *)                   FLOCN2
      CALL WLKBCK                                                  FLOCN2
      CALL EXIT                                                    FLOCN2
C                                                                  FLOCN2
   40 NTB = MLOC(1)                                                FLOCN2
      IC  = NLOC(1)                                                FLOCN2
      DO 200 I=1,IC,2                                      *NEW    FLOCN2
      NG    = NLOC(I+1)                                    **-1    FLOCN2
      ITUBE = NLOC(I+2)                                            FLOCN2
      IF(NG    .GT. NGT) GO TO 60                          **-1    FLOCN2
      IF(ITUBE .LE. NTB) GO TO 100                                 FLOCN2
   60 CALL TOPLIN                                                  FLOCN2
      WRITE(6,80) (NLOC(I+J),J=1,2),CP                     *NEW    FLOCN2
   80 FORMAT(60H0* * *  ERROR IN FLOW CONDUCTION DATA INPUT TO FLOCN2  *   **-1    FLOCN2
     1 * * // 5H0NG = I4, 8H ITUBE = I4, 5H CP = G13.3)            FLOCN2
      CALL WLKBCK                                                  FLOCN2
      CALL EXIT                                                    FLOCN2
C                                                                  FLOCN2
  100 NW = MLOC(ITUBE+1)                                           FLOCN2
      G(NG) = W*CP                                                 FLOCN2
  200 CONTINUE                                                     FLOCN2
C                                                                  FLOCN2
      RETURN                                                       FLOCN2
      END                                                          FLOCN2
```

```
FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT    FLPRNT

            SUBROUTINE FLPRNT(DATA,HEAD)                                                   FLPRNT
      C                                                                                    FLPRNT
            DIMENSION DATA(1), HEAD(9)                                                     FLPRNT
      C                                                                                    FLPRNT
            COMMON /FIXCON/ L(1)                                                           FLPRNT
      C                                                                                    FLPRNT
            EQUIVALENCE (NNT,D)                                                            FLPRNT
      C                                                                                    FLPRNT
      C                                                                                    FLPRNT
            D = DATA(1)                                                                    FLPRNT
            IF(L(29) .EQ. 0 .OR. L(28) .GE. 60) CALL TOPLIN                                FLPRNT
            WRITE(6,101) HEAD                                                              FLPRNT
            L(28) = L(28) + 2                                                              FLPRNT
            NS = 1                                                                         FLPRNT
            NF = 10                                                           *NEW         FLPRNT
          5 IF(NF .GT. NNT) GO TO 20                                          *NEW         FLPRNT
         10 WRITE(6,100) (DATA(I+1),I=NS,NF), NF                              *NEW         FLPRNT
            L(28) = L(28) + 1                                                 **-4         FLPRNT
            IF(L(28) .GE. 60) CALL TOPLIN                                                  FLPRNT
            IF(NF .EQ. NNT) RETURN                                                         FLPRNT
            NS = NF + 1                                                                    FLPRNT
            NF = NF + 10                                                      *NEW         FLPRNT
            GO TO 5                                                           **-1         FLPRNT
         20 WRITE(6,100) (DATA(I+1),I=NS,NNT)                                 *NEW         FLPRNT
            IF(L(28) .GE. 60) CALL TOPLIN                                     *NEW         FLPRNT
            RETURN                                                            *NEW         FLPRNT
        100 FORMAT(1X 5G12.5, 5X 5G12.5, I5)                                  *NEW         FLPRNT
        101 FORMAT(1H0 21A6, A5)                                              **-1         FLPRNT
            END                                                                            FLPRNT
```

```
      SUBROUTINE FLRES(L30,NTB)                                   FLRES
C                                                                 FLRES
      LOGICAL LRO, LMU, COP                                       FLRES
C                                                                 FLRES
      DIMENSION RDATA(1)                                          FLRES
C                                                                 FLRES
      COMMON /ARRAY/ NDATA(1)                                     FLRES
      COMMON /TEMP / T    (1)                                     FLRES
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9              FLRES
      COMMON /FDATA / LVP, LIFR, LAR, LOP                         FLRES
      COMMON /FDATA / COP, LRO, NRO, RO, LMU, NMU, XMU, GC2       FLRES
      COMMON /FDATA / TOL, MXPASS, EPS, FRDF                      FLRES
      COMMON/XSPACE/NDIM,NTH,NEXT(1)                              FLRES
      COMMON/POINTN/LNODE                                         FLRES
C                                                                 FLRES
      EQUIVALENCE (RDATA,NDATA), (HL,NHL)                         FLRES
C                                                                 FLRES
      DATA MAXI /65000/                                           FLRES
C                                                                 FLRES
C                                                                 FLRES
      WNTB = ABS(RDATA(L2+NTB) )                            *NEW  FLRES
      W4   = 4.0*WNTB                                       *NEW  FLRES
      RSUM = 0.0                                            **-1  FLRES
      IC  = NDATA(L30)                                            FLRES
C                                                                 FLRES
C FLUID LUMP LOOP                                                 FLRES
C                                                                 FLRES
      DO 200 I=1,IC,3                                             FLRES
      K = L30 + I                                                 FLRES
      NFL   = NDATA(K)                                            FLRES
      ITYPE = NDATA(K+1)                                          FLRES
      NTL   = NDATA(K+2)                                          FLRES
      LTYPE = L7 + ITYPE*6 - 6                                    FLRES
      WP  = RDATA(LTYPE+1)                                        FLRES
      CSA = RDATA(LTYPE+2)                                        FLRES
      FLL = RDATA(LTYPE+3)                                        FLRES
      MFF = NDATA(LTYPE+4)                                        FLRES
      NHL = NDATA(LTYPE+5)                                        FLRES
      FFC = RDATA(LTYPE+6)                                        FLRES
C                                                                 FLRES
      IF(LRO) CALL DIDEGI(T(NFL),NDATA(NRO), RO)                  FLRES
      IF(LMU) CALL DIDEGI(T(NFL),NDATA(NMU),XMU)                  FLRES
      RE = W4/XMU/WP                                              FLRES
      IF(NHL .GT. 0 .AND. NHL .LT. MAXI) CALL DIDEGI(RE,NDATA(NHL),HL)  FLRES
      IF(RE .GT. 2000.0) GO TO 100                                FLRES
      WMU = XMU                                                   FLRES
      IF(LMU) CALL DIDEGI(T(NTL),NDATA(NMU),WMU)                  FLRES
      FF = 64.0/RE*SQRT(WMU/XMU)                                  FLRES
      GO TO 140                                                   FLRES
  100 IF(MFF .EQ. 0) GO TO 120                                    FLRES
      CALL DIDEGI(RE,NDATA(MFF),FF)                               FLRES
      GO TO 160                                                   FLRES
  120 IF(RE .LT. 4000.0) GO TO 140                                FLRES
      FF = 0.316/SQRT(SQRT(RE))                                   FLRES
      GO TO 160                                                   FLRES
  140 FF = 0.2086082052 + RE*(-0.1868265324E-3 + RE*(7.6236703785E-7  FLRES
     1   + RE*(-0.65545818E-11)))                                 FLRES
  160 R = (FF*FFC*FLL/(4.0*CSA/WP)+HL)*WNTB/GC2/CSA/CSA/RO  *NEW  FLRES
      RSUM = RSUM + R                                       **-1  FLRES
```

```
C                                                                FLRES
      IF(.NOT. COP) GO TO 200                                    FLRES
      CALL LINECK(5)                                             FLRES
      WRITE(6,180)NEXT(LNODE+NFL),T(NFL),ITYPE,NEXT(LNODE+NTL),T(NTL),  FLRES
     1(RDATA(LTYPE+J), J=1,4),FFC,HL,RO,XMU,RE,FF,R              FLRES
  180 FORMAT(/             7X 7HNFL    = I10  , 8X 7HT(NFL)= G13.8,  FLRES
     1  5X 7HITYPE = I10  , 8X 7HNTL    = I10  , 8X 7HT(NTL)= G13.8/  FLRES
     2  7X 7HWP    = G13.8, 5X 7HCSA    = G13.8, 5X 7HFLL   = G13.8,  FLRES
     3  5X 7HMFF   = I10  , 8X 7HFFC    = G13.8/ 7X 7HHL    = G13.8,  FLRES
     4  5X 7HRO    = G13.8, 5X 7HMU     = G13.8, 5X 7HRE    = G13.8,  FLRES
     5  5X 7HFF    = G13.8/ 7X 7HR      = G13.8 )                     FLRES
  200 CONTINUE                                                   FLRES
      RDATA(L4+NTB) = 1.0/RSUM                                   FLRES
C                                                                FLRES
      IF(.NOT. COP) GO TO 300                                    FLRES
      CALL LINECK (2)                                            FLRES
      WRITE(6,220) RDATA(L4+NTB)                                 FLRES
  220 FORMAT(/ 7X 7HG(NTB)= G13.8)                               FLRES
C                                                                FLRES
  300 RETURN                                                     FLRES
C                                                                FLRES
      END                                                        FLRES
```

```
      SUBROUTINE FLUX(NFLXTP,DATA,NCRV,DQTIME,QTIME)                    FLUX
C                                                                       FLUX
      DIMENSION DATA(1)                                                 FLUX
C                                                                       FLUX
      COMMON /FLXCON/ TIMEN                                             FLUX
C                                                                       FLUX
      EQUIVALENCE (D,N)                                                 FLUX
C                                                                       FLUX
C                                                                       FLUX
      IF(QTIME .GE. TIMEN) RETURN                                       FLUX
      IF(NFLXTP .GT. 0) READ(NFLXTP) FLXTIM                             FLUX
      NFLXTP = IABS(NFLXTP)                                             FLUX
   10 READ(NFLXTP) (NP, (DATA(I+2*J*NP+2*J-2*NP-1),I=1,NP,2),           FLUX
     1                    (DATA(I+2*J*NP+2*J-2*NP-1),I=2,NP,2),J=1,NCRV) FLUX
      READ(NFLXTP) FLXTIM                                               FLUX
      QTIME = FLXTIM + DQTIME                                           FLUX
      IF(QTIME .LE. TIMEN) GO TO 10                                     FLUX
      WRITE(6,20) QTIME                                                 FLUX
   20 FORMAT(22H FLUX TABLES ENDING AT  G11.5, 15H HAVE BEEN READ )     FLUX
      LOC = 1                                                           FLUX
      D = DATA(LOC)                                                     FLUX
      IC = N                                                            FLUX
      DO 30 J=1,IC,2                                                    FLUX
      DATA(LOC+J) = DATA(LOC+J) + DQTIME                                FLUX
   30 CONTINUE                                                          FLUX
      LOC = LOC + IC + 1                                                FLUX
   40 CONTINUE                                                          FLUX
      NFLXTP = -NFLXTP                                                  FLUX
      RETURN                                                            FLUX
      END                                                               FLUX
```

GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT GENOUT

```
      SUBROUTINE GENOUT(NDATA,ISTRT,ISTP,NAME)                                     GENOUT
C                                                                                  GENOUT
      DIMENSION FMT(12), NAME(22)                                                  GENOUT
      DIMENSION NDATA(1)                                                           GENOUT
      DATA MAXI /   134217728/                                                     GENOUT
      DATA FMT(1), FMT(12) / 6H(1X1P  , 6H I10)  /                                 GENOUT
      LOGICAL ONE, CKD                                                             GENOUT
C                                                                                  GENOUT
      BASE = 6HE12.4,                                                              GENOUT
      ASSIGN 32 TO MM                                                              GENOUT
      CKD = .FALSE.                                                                GENOUT
      GO TO 5                                                                      GENOUT
      ENTRY GENI(NDATA,ISTRT,ISTP,NAME)                                            GENOUT
      BASE = 6HI9, 3X                                                              GENOUT
      ASSIGN 45 TO MM                                                              GENOUT
      CKD = .TRUE.                                                                 GENOUT
      GO TO 5                                                                      GENOUT
      ENTRY GENR(NDATA,ISTRT,ISTP,NAME)                                            GENOUT
      BASE = 6HE12.4,                                                              GENOUT
      ASSIGN 45 TO MM                                                              GENOUT
      CKD = .TRUE.                                                                 GENOUT
    5 WRITE(6,10) NAME                                                             GENOUT
   10 FORMAT(22A6)                                                                 GENOUT
      ONE = .FALSE.                                                                GENOUT
      IF(ISTRT .EQ. 1 .AND. ISTP .EQ. 1) ONE = .TRUE.                              GENOUT
   15 I=ISTRT                                                                      GENOUT
   20 IF(I .GT. ISTP) GO TO 70                                                     GENOUT
      CALL LINECK(1)                                                     *NEW      GENOUT
      L=1                                                                          GENOUT
      DO 30 J=2,11                                                                 GENOUT
      FMT(J) = BASE                                                                GENOUT
   30 CONTINUE                                                                     GENOUT
      IF(CKD .AND. ONE) GO TO 60                                                   GENOUT
      IF(I .NE. ISTRT) GO TO 36                                                    GENOUT
      M=ISTRT-10*(ISTRT/10)                                                        GENOUT
      IF(M .EQ. 1) GO TO 36                                                        GENOUT
      IF(M .EQ. 0) M = 10                                                          GENOUT
      DO 35 J=2,M                                                                  GENOUT
      FMT(J)=6H 12X                                                                GENOUT
   35 CONTINUE                                                                     GENOUT
      L=M                                                                          GENOUT
      J=I-M+10                                                                     GENOUT
      GO TO 38                                                                     GENOUT
   36 J=I+9                                                                        GENOUT
   38 IF(J .LE. ISTP) GO TO 39                                                     GENOUT
      M = ISTP - J + 12                                                            GENOUT
      J=ISTP                                                                       GENOUT
      DO 37 K=M,11                                                                 GENOUT
      FMT(K)=6H 12X                                                                GENOUT
   37 CONTINUE                                                                     GENOUT
   39 GO TO MM, (32,45)                                                            GENOUT
   32 DO 40 K=I,J                                                                  GENOUT
      L=L+1                                                                        GENOUT
      IF(IABS(NDATA(K)) .LE. MAXI)   FMT(L)=6HI9, 3X                               GENOUT
   40 CONTINUE                                                                     GENOUT
   45 IF(ONE) GO TO 60                                                             GENOUT
      WRITE(6,FMT)(NDATA(K),K=I,J),J                                               GENOUT
      I=J+1                                                                        GENOUT
      GO TO 20                                                                     GENOUT
```

```
60 WRITE(6,FMT) NDATA(ISTRT)                                    GENOUT
70 RETURN                                                        GENOUT
   END                                                           GENOUT
```

```
      DIMENSION NX(1), MVF(26), NVF(11), NFC(3)                                        GEPLOT
      DIMENSION YLO(75),YHI(75),ORD(1),BUFR(4000),XY(33000)                            GEPLOT
      INTEGER  TITLEA(12),TITLEB(20),TITLEC(20),TITLES(9,75)          GEPLT003         GEPLOT
     1 ,ITYTLS(9,75),BCDX(4),BCDY(4,11),ITEM(75),ITYPE(75)            GEPLT004         GEPLOT
     2 ,IGS(76),KEYA(11),KEYB(12),BLANK                               GEPLT005         GEPLOT
     3 ,TMESCL(3)                                                                      GEPLOT
      DIMENSION  LOC(76),ABS(1)                                       GEPLT006         GEPLOT
      DIMENSION ITMAVG(50), AVG(150), AVGLOC(100), HDR(12)                             GEPLOT
C                                                                     GEPLT007         GEPLOT
      COMMON  NPTS,TPG,BUFR                                           GEPLT008         GEPLOT
      COMMON /XYARY/ XY                                                                GEPLOT
      EQUIVALENCE (BUFR(1),ABS(1)),(BUFR(2001),ORD(1)),               GEPLT009         GEPLOT
     1            (TITLES(1,1),ITYTLS(1,1))                           GEPLT010         GEPLOT
      EQUIVALENCE (NX,XY)                                                              GEPLOT
C                                                                     GEPLT011         GEPLOT
C               ARRAY DEFINITIONS                                     GEPLT012         GEPLOT
C                                                                     GEPLT013         GEPLOT
C ABS    - ABSCISSA VALUES FOR THE CURRENT FRAME                      GEPLT014         GEPLOT
C BCDX   - ITEM NAMES AND DIMENSION INFORMATION ON THE ABCISSA        GEPLT015         GEPLOT
C BCDY   - ITEM NAMES AND DIMENSION INFORMATION ON THE ORDINATE       GEPLT016         GEPLOT
C BUFR   - BUFFER FOR READING HISTORY TAPE RECORDS                    GEPLT017         GEPLOT
C IGS    - ARRAY FOR STORING THE ITEM TYPE INDICES                    GEPLT018         GEPLOT
C ITEM   - THE ITEM NUMBERS TO BE PLOTTED                             GEPLT019         GEPLOT
C ITYPE  - THE ITEM TYPES FOR THE RESPECTIVE ITEM NUMBERS             GEPLT020         GEPLOT
C KEYA   - ITEM TYPE CODE ARRAY                                       GEPLT021         GEPLOT
C KEYB   - INDEX TO ITEM TYPE IN BUFR ARRAY                           GEPLT022         GEPLOT
C LOC    - INDEX TO ITEM ON EACH TIME RECORD (ERROR CODE IF NEGATIVE) GEPLT023         GEPLOT
C ORD    - ORDINATE VALUES FOR THE CURRENT FRAME                      GEPLT024         GEPLOT
C TITLEA - GENERAL TITLE FOR EACH FRAME                               GEPLT025         GEPLOT
C TITLEB - TITLE OF 1-ST AND 2-ND ITEMS ON THE CURRENT FRAME          GEPLT026         GEPLOT
C TITLEC - TITLE OF 3-RD AND 4-TH ITEMS ON THE CURRENT FRAME          GEPLT027         GEPLOT
C TITLES - THE ITEM PLOTTING SYMBOLS AND DESCRIPTIONS                 GEPLT028         GEPLOT
C XY     - ARRAY FOR ITEMS TO BE PLOTTED (INCLUDING TIME)             GEPLT029         GEPLOT
C YHI    - THE MAXIMUM ORDINATE VALUES                                GEPLT030         GEPLOT
C YLO    - THE MINIMUM ORDINATE VALUES                                GEPLT031         GEPLOT
C                                                                     GEPLT032         GEPLOT
C               WORD DEFINITIONS                                      GEPLT033         GEPLOT
C                                                                     GEPLT034         GEPLOT
C ITEMS  - THE NUMBER OF ITEMS PER TIME RECORD FOR PLOTTING - MAX = 75 GEPLT035        GEPLOT
C NGRDS  - THE NUMBER OF GRIDS REQUIRED TO SPAN THE RANGE (TZ - TA)   GEPLT036         GEPLOT
C NSIZE  - THE NUMBER OF WORDS ALLOTTED TO THE XY ARRAY               GEPLT037         GEPLOT
C NTOTL  - NUMBER OF WORDS USED IN THE XY ARRAY                       GEPLT038         GEPLOT
C NTYMS  - THE NUMBER OF POINTS TO BE PLOTTED ( = NSIZE/ITEMS)        GEPLT039         GEPLOT
C NWRDS  - THE NUMBER OF ITEMS PER TIME RECORD ON THE HISTORY TAPE    GEPLT040         GEPLOT
C                                                                     GEPLT041         GEPLOT
C               INITIALIZATION                                        GEPLT042         GEPLOT
C                                                                     GEPLT043         GEPLOT
      DATA (BCDX(I),I=1,4) /24H     TIME - (*****)      /                              GEPLOT
      DATA (TMESCL(I),I=1,3) /18H( SEC ( MIN (HOURS/                                   GEPLOT
      DATA NVF/'(38X,6HLOADED,F11.5,22H ******    LOOKING FOR,F11.5,6H**               GEPLOT
     1****)'/                                                                          GEPLOT
      DATA NFC/' SEC.  MIN.  HRS. '/                                                   GEPLOT
      DATA MVF/'(1H1,48X,23HP L O T   P R O G R A M//24X,8HTITLE  -,2X,                GEPLOT
     112A6///28X,5HFROM ,F10.3,10H****** TO  , F10.3,  13H******, WITH ,               GEPLOT
     2F10.3, 15H****** PER GRID/////  )'/                                              GEPLOT
C                                                                     GEPLT045         GEPLOT
      DATA ((BCDY(I,J),I=1,4),J=1,11) /                                                GEPLOT
     1 24H                        ,                                                    GEPLOT
     2 24H                        ,                                                    GEPLOT
```

```
     3 24H         PRESSURE          ,                                        GEPLOT
     4 24H  VALVE POSITION DATA      ,                                        GEPLOT
     5 24H                           ,                                        GEPLOT
     6 24H                           ,                                        GEPLOT
     7 24H                           ,                                        GEPLOT
     8 24H          FLOW RATE        ,                                        GEPLOT
     9 24H        TEMPERATURE        ,                                        GEPLOT
     X 24H        TEMPERATURE        ,                                        GEPLOT
     1 24H        TEMPERATURE         /                                       GEPLOT
C                                                                    GEPLT052 GEPLOT
C                                                    ITEM TYPE CODESGEPLT053 GEPLOT
      DATA KEYA / 2HBU, 2HOH, 2HXX, 2HVP, 2HMT, 2HLB,                GEPLT054 GEPLOT
     1              2HBT, 2HFR, 2HFT, 2HTT, 2HST       /             GEPLT055 GEPLOT
C                                                                    GEPLT056 GEPLOT
      DATA BLANK /6H      /                                          GEPLT057 GEPLOT
C                                                                    GEPLT058 GEPLOT
      DATA D1, D2 /0007700000000, 0050005050505 /                             GEPLOT
C                                                                    GEPLT108 GEPLOT
      INTEGER  PSYM(4)                                               GEPLT109 GEPLOT
      DATA  (PSYM(I),I=1,4)/24HE1]   [2]   [3]   [4]   /             GEPLT110 GEPLOT
      EXTERNAL  TABLIV                                               GEPLT111 GEPLOT
      CALL CHSIZV(2,2)                                               GEPLT112 GEPLOT
      CALL RITSTV(12,18,TABLIV)                                      GEPLT113 GEPLOT
C                                                                    GEPLT114 GEPLOT
      CALL RESET                                                     GEPLT115 GEPLOT
      NCASE = 1                                                      GEPLT116 GEPLOT
      NSIZE=33000                                                             GEPLOT
      GO TO 60                                                       GEPLT118 GEPLOT
   20 CALL CLOCK(ETIME)                                              GEPLT119 GEPLOT
      WRITE (6,40) ETIME                                             GEPLT120 GEPLOT
   40 FORMAT(////11X,'COMPUTER TIME = ',F10.5,'   MINUTES')          GEPLT121 GEPLOT
C                                    READ THE CASE CARDS AND PRINT THE HEADINGGEPLT122 GEPLOT
   60 READ(5,80,END=160) TITLEA, TA, TZ, TPG, ITMX, MPNT, NTP, KT, INC,       GEPLOT
     1 IUNIT, ASTAT, ASTOP                                                    GEPLOT
   80 FORMAT(12A6/3F10.0, 6I5, 2F10.0)                                        GEPLOT
      IF(KT .LT. 1) KT = 23                                                   GEPLOT
C                                              CHECK FOR COMBINE OPTIONGEPLT126 GEPLOT
      IF (NTP.EQ.0) GO TO 100                                        GEPLT127 GEPLOT
      CALL COMBIN(NTP,KT,INC,IUNIT)                                  GEPLT128 GEPLOT
      CALL CLOCK(ETIME)                                              GEPLT129 GEPLOT
      WRITE (6,40) ETIME                                             GEPLT130 GEPLOT
  100 INSTRY=KT                                                      GEPLT131 GEPLOT
C                                                                             GEPLOT
C                                           CHECK FOR BLANK - END OF JOBGEPLT138 GEPLOT
      IF (TPG) 160,160,180                                           GEPLT139 GEPLOT
  160 CALL EXIT                                                      GEPLT140 GEPLOT
C                                                                    GEPLT141 GEPLOT
  180 IF(ITMX .LT. 1 .OR. ITMX .GT. 3) GO TO 200                              GEPLOT
      BCDX(3) = TMESCL(ITMX)                                                  GEPLOT
      MVF( 5) = NFC(ITMX)                                                     GEPLOT
      MVF(10) = NFC(ITMX)                                                     GEPLOT
      MVF(15) = NFC(ITMX)                                                     GEPLOT
      MVF(19) = NFC(ITMX)                                                     GEPLOT
      MVF(23) = NFC(ITMX)                                                     GEPLOT
  200 WRITE(6,MVF) TITLEA, TA, TZ, TPG                                        GEPLOT
C                                                                    GEPLT154 GEPLOT
C                                 READ AND PRINT THE HISTORY TAPE HEADER LABELGEPLT155 GEPLOT
  240 READ(INSTRY) HDR, (LOC(I),I=13,26), NSL, (NX(I),I=1,NSL)                GEPLOT
      LOC(27) = NSL                                                           GEPLOT
      NSIZE = NSIZE - NSL                                                     GEPLOT
      LNODE = NSIZE                                                           GEPLOT
      DO 250 I=1,NSL                                                          GEPLOT
      NX(LNODE+I) = NX(I)                                                     GEPLOT
```

```
      250 CONTINUE                                                      GEPLOT
C                      INDEX AND COUNT THE ITEMS ON THE HISTORY TAPEGEPLT157  GEPLOT
         KEYB(1) = 1                                                GEPLT158  GEPLOT
         DO 260 I=2,12                                              GEPLT159  GEPLOT
  260 KEYB(I) = KEYB(I-1) + LOC(I+15)                               GEPLT160  GEPLOT
         NWRDS = KEYB(12)                                           GEPLT161  GEPLOT
         IF (NCASE.NE.1) GO TO 300                                  GEPLT162  GEPLOT
         WRITE(6,280) HDR, (LOC(I+16),KEYA(I),I=1,11)                         GEPLOT
  280 FORMAT(52X,25HTHE HISTORY TAPE LABEL IS//29X,12A6///18X,22HTHE ITEGEPLT164  GEPLOT
     1M COUNTS ARE - , 6(I6,A2) / 40X 5(I6,A2)/////)                GEPLT165  GEPLOT
C                                                                   GEPLT166  GEPLOT
  300 IFINIS = 0                                                    GEPLT167  GEPLOT
      JFINIS = 0                                                    GEPLT168  GEPLOT
C                                          READ THE ITEMS TO BE PLOTTEDGEPLT172  GEPLOT
         ITEMS = 76                                                 GEPLT173  GEPLOT
         I = 1                                                      GEPLT174  GEPLOT
         J = 0                                                      GEPLT175  GEPLOT
         NOAVG = 0                                                            GEPLOT
         KSW = 0                                                              GEPLOT
  320 READ(5,340) ITEM(I),ITYPE(I),IREL,KAVG,                           *NEW  GEPLOT
     1 (TITLES(J,I),J=2,9), YLO(I), YHI(I)                              **-1  GEPLOT
  340 FORMAT(I5,A2,I1,I2,8A6,2X2F10.0)                                  *NEW  GEPLOT
C                                      TEST FOR END OF JOB  -  BLANK CARDGEPLT179**-1  GEPLOT
         IF (ITEM(1) .EQ. 0) GO TO 20                               GEPLT180  GEPLOT
         IF (ITEM(I) .EQ. 0) GO TO 360                              GEPLT181  GEPLOT
C                                                                             GEPLOT
         IF(ITYPE(I) .NE. KEYA(11)) GO TO 344                                 GEPLOT
         IF(IREL .NE. 0) GO TO 344                                      *NEW  GEPLOT
         IACT = IABS(ITEM(I))                                           **-3  GEPLOT
         DO 341 L=1,NSL                                                       GEPLOT
         IF(IACT .EQ. NX(LNODE+L)) GO TO 342                                  GEPLOT
  341 CONTINUE                                                                GEPLOT
         GO TO 344                                                            GEPLOT
  342 ITEM(I) = ISIGN(L,ITEM(I))                                              GEPLOT
  344 CONTINUE                                                                GEPLOT
C                                  CHECK FOR NEW GRID SET SPECIFIED BY USERGEPLT182  GEPLOT
         IF (ITEM(I) .LT. 0) J = 0                                  GEPLT183  GEPLOT
         IF (ITEM(I) .LT. 0) KSW = 0                                          GEPLOT
C                                PUT BCD PLOTTING SYMBOL INTO TITLES ARRAYGEPLT184  GEPLOT
         J = J+1                                                    GEPLT185  GEPLOT
         ITYTLS(1,I) = BLANK                                        GEPLT186  GEPLOT
         FLD(30,6,ITYTLS(1,I)) = J + KSW + 48                                 GEPLOT
         IF(KAVG .EQ. 0 .OR. NOAVG .GE. 50) GO TO 345                         GEPLOT
         NOAVG = NOAVG + 1                                                    GEPLOT
         ITMAVG(NOAVG) = I                                                    GEPLOT
         IF(KAVG .LT. 10) GO TO 345                                           GEPLOT
         ITMAVG(NOAVG) =-I                                                    GEPLOT
         FLD(0,30,ITYTLS(1,I)) = 6H YES                                       GEPLOT
         KSW = KSW + 1                                                        GEPLOT
C                     BUMP ITEM COUNTER AND CHECK FOR MAXIMUM NUMBER OF ITEMSGEPLT188  GEPLOT
  345 I = I + 1                                                               GEPLOT
         IF(I+1 .LT. ITEMS) GO TO 320                                         GEPLOT
  360 ITEMS = I                                                     GEPLT191  GEPLOT
         DO 370 L=1,NOAVG                                                     GEPLOT
         AVG(L) = 0.                                                          GEPLOT
  370 CONTINUE                                                                GEPLOT
C                                          SET FIRST ITEM FOR NEW GRID SETGEPLT192  GEPLOT
         ITEM(1) = -IABS(ITEM(1))                                   GEPLT193  GEPLOT
C                                     FIND THE TYPE CODE IN THE KEYA ARRAYGEPLT194  GEPLOT
  380 LOC(1) = 1                                                    GEPLT195  GEPLOT
         I = 1                                                      GEPLT196  GEPLOT
  400 J = 11                                                        GEPLT197  GEPLOT
         K = I+1                                                    GEPLT198  GEPLOT
```

```
  420 IF (ITYPE(I) .EQ. KEYA(J)) GO TO 460                       GEPLT199   GEPLOT
      J = J-1                                                    GEPLT200   GEPLOT
      IF (J .GT. 0) GO TO 420                                    GEPLT201   GEPLOT
C                                           INCORRECT TYPE CODEGEPLT202     GEPLOT
      LOC(K) = -1                                                GEPLT203   GEPLOT
      IACT = ITEM(I)                                                        GEPLOT
      IF(ITYPE(I) .NE. KEYA(11)) GO TO 430                                  GEPLOT
      NN = IABS(IACT)                                                       GEPLOT
      IACT = ISIGN(NX(LNODE+NN),ITEM(I))                                    GEPLOT
  430 WRITE(6,440) IACT, ITYPE(I)                                           GEPLOT
  440 FORMAT(48X,4HITEM,I8,A2,20H  TYPE CODE IN ERROR)           GEPLT205   GEPLOT
      GO TO 540                                                  GEPLT206   GEPLOT
C                              CHECK THE MAGNITUDE OF THE ITEM NUMBERGEPLT207   GEPLOT
  460 IF ((KEYB(J+1) - KEYB(J)) - IABS(ITEM(I))) 480,520,520     GEPLT208   GEPLOT
C                                         INCORRECT ITEM NUMBERGEPLT209     GEPLOT
  480 LOC(K) = -2                                                GEPLT210   GEPLOT
      IACT = ITEM(I)                                                        GEPLOT
      IF(ITYPE(I) .NE. KEYA(11)) GO TO 490                                  GEPLOT
      NN = IABS(IACT)                                                       GEPLOT
      IACT = ISIGN(NX(LNODE+NN),ITEM(I))                                    GEPLOT
  490 WRITE(6,500) IACT, ITYPE(I)                                           GEPLOT
  500 FORMAT (49X,4HITEM,I7,A2,17H  IS OUT OF RANGE)             GEPLT212   GEPLOT
      GO TO 540                                                  GEPLT213   GEPLOT
C                                                                GEPLT214   GEPLOT
  520 LOC(K) = KEYB(J)+IABS(ITEM(I))                             GEPLT215   GEPLOT
C                                     SAVE FUNCTION TYPE INDEXGEPLT216      GEPLOT
  540 IGS(I) = J                                                 GEPLT217   GEPLOT
C                             BUMP ITEM NUMBER AND TEST FOR LAST ITEMGEPLT219   GEPLOT
      I = I+1                                                    GEPLT220   GEPLOT
      IF (I .LT. ITEMS) GO TO 400                                GEPLT221   GEPLOT
C                                                                GEPLT222   GEPLOT
C                          START LOADING THE DATA FROM THE HISTORY TAPEGEPLT223   GEPLOT
C                                                                GEPLT224   GEPLOT
C                              COMPUTE THE MAXIMUM NUMBER OF RECORDSGEPLT225   GEPLOT
  560 NTYMS = NSIZE/ITEMS                                        GEPLT226   GEPLOT
      WRITE (6,580)                                              GEPLT227   GEPLOT
  580 FORMAT(1H1,44X,40HPOSITIONING AND READING THE HISTORY TAPE/)   GEPLT228   GEPLOT
C                                         POSITION THE HISTORY TAPEGEPLT229   GEPLOT
      NTPTS = 0                                                             GEPLOT
      I = 1                                                      GEPLT230   GEPLOT
      J = 1                                                      GEPLT231   GEPLOT
  600 READ (IHSTRY) (BUFR(L),L=1,NWRDS)                          GEPLT232   GEPLOT
C                                           CHECK FOR END OF DATA FILEGEPLT233   GEPLOT
      IF (BUFR(1).LT.0.0) GO TO 780                              GEPLT234   GEPLOT
C                                      CHECK FOR REQUESTED START TIMEGEPLT235   GEPLOT
      IF (BUFR(1).LT.TA) GO TO 620                               GEPLT236   GEPLOT
      GO TO 660                                                  GEPLT237   GEPLOT
  620 IF(MPNT .EQ. 1) WRITE(6,NVF) BUFR(1), TA                              GEPLOT
      GO TO 600                                                             GEPLOT
  660 IF(MPNT .EQ. 1) WRITE(6,NVF) BUFR(1), TZ                              GEPLOT
C                                      CHECK FOR REQUESTED FINAL TIMEGEPLT242   GEPLOT
      IF (BUFR(1) .GT. TZ) IFINIS = 1                            GEPLT243   GEPLOT
C                              PICK UP THE ITEM/ITYPE ARRAY QUANTITIESGEPLT244   GEPLOT
      DO 740 L=1,ITEMS                                           GEPLT245   GEPLOT
      M = LOC(L)                                                 GEPLT246   GEPLOT
C                                             CHECK FOR ERROR ITEMGEPLT247   GEPLOT
      IF (M .LT. 0) GO TO 740                                    GEPLT248   GEPLOT
  680 XY(J) = BUFR(M)                                            GEPLT255   GEPLOT
C                                       BUMP THE XY ARRAY SUBSCRIPTGEPLT262   GEPLOT
  740 J = J+1                                                    GEPLT263   GEPLOT
      IF(NOAVG .EQ. 0) GO TO 752                                            GEPLOT
      IF(BUFR(1) .LT. ASTRT-.0005 .OR. BUFR(1) .GT. ASTOP+.0005)            GEPLOT
     X   GO TO 752                                                          GEPLOT
```

```
      NTPTS = NTPTS + 1                                                         GEPLOT
      STOP = BUFR(I)                                                            GEPLOT
      DO 749 L=1,NOAVG                                                          GEPLOT
      MM = IABS(ITMAVG(L))                                                      GEPLOT
      M = LOC(MM+1)                                                             GEPLOT
      IF(M .LT. 0) GO TO 749                                                    GEPLOT
      M = MM + J - ITEMS                                                        GEPLOT
      AVG(L) = AVG(L) + XY(M)                                                   GEPLOT
      IF(NTPTS .GT. 1) GO TO 743                                                GEPLOT
      ISTART = I - 1                                                            GEPLOT
      START = BUFR(I)                                                           GEPLOT
      AVGLOC(L    ) = BUFR(I)                                                   GEPLOT
      AVGLOC(L+ 50) = BUFR(I)                                                   GEPLOT
      AVG   (L+ 50) = XY(M)                                                     GEPLOT
      AVG   (L+100) = XY(M)                                                     GEPLOT
      GO TO 749                                                                 GEPLOT
  743 IF(XY(M) .LE. AVG(L+ 50)) GO TO 746                                       GEPLOT
      AVGLOC(L    ) = BUFR(I)                                                   GEPLOT
      AVG   (L+ 50) = XY(M)                                                     GEPLOT
      GO TO 749                                                                 GEPLOT
  746 IF(XY(M) .GE. AVG(L+100)) GO TO 749                                       GEPLOT
      AVGLOC(L+ 50) = BUFR(I)                                                   GEPLOT
      AVG   (L+100) = XY(M)                                                     GEPLOT
  749 CONTINUE                                                                  GEPLOT
  752 LJ = J - ITEMS + 1                                                        GEPLOT
      LJI = J - 1                                                       GEPLT265  GEPLOT
      IF(MPNT .EQ. 1) WRITE(6,760) (XY(L),L=LJ,LJI)                               GEPLOT
  760 FORMAT (10F11.3)                                                  GEPLT267  GEPLOT
      IF (IFINIS .EQ. 1) GO TO 800                                      GEPLT268  GEPLOT
      I = I+1                                                           GEPLT269  GEPLOT
C                                      CHECK FOR MAXIMUM NUMBER OF POINTSGEPLT270  GEPLOT
      IF (I .LE. NTYMS) GO TO 600                                       GEPLT271  GEPLOT
  780 NTYMS = I-1                                                       GEPLT272  GEPLOT
      GO TO 820                                                         GEPLT273  GEPLOT
  800 NTYMS = I                                                         GEPLT274  GEPLOT
C                           COMPUTE THE NUMBER OF WORDS USED IN THE XY ARRAYGEPLT275  GEPLOT
  820 NTOTL = J-1                                                       GEPLT276  GEPLOT
      REWIND INSTAY                                                     GEPLT277  GEPLOT
      WRITE(6,840) ITEMS, I, NTOTL                                      GEPLT278  GEPLOT
  840 FORMAT (1H0,I10,42H  DATA VALUES HAVE BEEN STORED FOR EACH OF,I6, GEPLT279  GEPLOT
     1 13H  TIME POINTS/1XI10,30H  DATA VALUES HAVE BEEN STORED)        GEPLT280  GEPLOT
C                                                                                 GEPLOT
      IF(NTPTS .EQ. 0) GO TO 852                                                  GEPLOT
      WRITE(6,843) HDR, NTPTS, ASTRT, ASTOP, START, STOP                          GEPLOT
  843 FORMAT(1H1 12A6/                                                            GEPLOT
     X '0THE NUMERICAL AVERAGES FOR THE FOLLOWING ITEMS WERE REQUESTED'           GEPLOT
     X ' FOR THE' I4,' TIME POINTS'/' BEGINNING WITH'F7.3,                        GEPLOT
     X ' HRS., AND ENDING WITH' F7.3,' HRS.'/' ACTUAL TIMES -' F7.3,              GEPLOT
     X ' HRS., AND             ' F7.3,' HRS.'//'  ITEM  TYPE     DESCRIPT         GEPLOT
     XION' 41X 'AVERAGE'7X'MAX VALUE'5X'TIME'7X'MIN VALUE'5X'TIME'/)              GEPLOT
      TPTS = NTPTS                                                                GEPLOT
      DO 849 L=1,NOAVG                                                            GEPLOT
      M = IABS(ITMAVG(L))                                                         GEPLOT
      MM = LOC(M+1)                                                               GEPLOT
      IF(MM .LT. 0) GO TO 849                                                     GEPLOT
      AVG(L) = AVG(L)/TPTS                                                        GEPLOT
      IACT = ITEM(M)                                                              GEPLOT
      IF(ITYPE(M) .NE. KEYA(11)) GO TO 845                                        GEPLOT
      NN = IABS(IACT)                                                             GEPLOT
      IACT = ISIGN(NX(LNODE+NN), ITEM(M))                                         GEPLOT
  845 WRITE(6,846) IACT, ITYPE(M), (TITLES(J,M),J=2,9), AVG(L),                   GEPLOT
     X AVG(L+50), AVGLOC(L), AVG(L+100), AVGLOC(L+50)                             GEPLOT
  846 FORMAT(1XI5, 4XA2, 5X8A6, 1XF10.2, 2(6XF10.2, 2XF7.3))                      GEPLOT
```

```
  849 CONTINUE                                                          GEPLOT
  852 CONTINUE                                                          GEPLOT
C                                  FIND THE MAXIMUM AND MINIMUM ORDINATESGEPLT404 GEPLOT
  920 I = 1                                                     GEPLT405 GEPLOT
  940 J = I                                                     GEPLT406 GEPLOT
C                                              PICK UP THE INPUT VALUESGEPLT407 GEPLOT
      YB = YLO(I)                                               GEPLT408 GEPLOT
      YT = YHI(I)                                               GEPLT409 GEPLOT
      LYS = -1                                                  GEPLT410 GEPLOT
      IF (YT-YB) 960,960,980                                    GEPLT411 GEPLOT
  960 YB = 1.E10                                                GEPLT412 GEPLOT
      YT = -1.E10                                               GEPLT413 GEPLOT
  980 K = I+1                                                   GEPLT414 GEPLOT
C                                                 CHECK FOR ERROR ITEMGEPLT415 GEPLOT
      IF (LOC(K) .LT. 0) GO TO 1020                             GEPLT416 GEPLOT
      IF ((YHI(I)-YLO(I)).GT.0.01) LYS = 1                      GEPLT417 GEPLOT
C                                       COMPARE WITH THE TAPE VALUESGEPLT418 GEPLOT
      DO 1000 L=K,NTOTL,ITEMS                                   GEPLT419 GEPLOT
      YB = AMIN1(XY(L),YB)                                      GEPLT420 GEPLOT
 1000 YT = AMAX1(XY(L),YT)                                      GEPLT421 GEPLOT
C                                                  CHECK FOR LAST ITEMGEPLT422 GEPLOT
 1020 I = I+1                                                   GEPLT423 GEPLOT
      IF (I .LT. ITEMS) GO TO 1040                              GEPLT424 GEPLOT
      JFINIS = 1                                                GEPLT425 GEPLOT
      GO TO 1060                                                GEPLT426 GEPLOT
C                                              CHECK FOR NEW GRID SETGEPLT427 GEPLOT
 1040 IF (ITEM(I) .GT. 0) GO TO 980                             GEPLT428 GEPLOT
 1060 YLO(J) = YB                                                        GEPLOT
      YHI(J) = YT                                               GEPLT435 GEPLOT
      IF (JFINIS .EQ. 0) GO TO 940                              GEPLT436 GEPLOT
C                                       PRINT THE ITEMS TO BE PLOTTEDGEPLT437 GEPLOT
      WRITE (6,1100)                                            GEPLT438 GEPLOT
 1100 FORMAT(1H1 14X 'ITEM TYPE' 6X 'AVG  PLOTTING SYMBOL AND'           GEPLOT
     X ' DESCRIPTION' 28X 'Y-MIN      Y-MAX     STATUS'/)                GEPLOT
      JJ = ITEMS - 1                                                     GEPLOT
      DO 1200 I=1,JJ                                                     GEPLOT
 1120 WRITE(6,1140) I,ITEM(I),ITYPE(I),(TITLES(J,I),J=1,9),YLO(I),      GEPLOT
     1                 YHI(I), LOC(I+1)                         GEPLT446 GEPLOT
 1140 FORMAT(4XI5,5XI5,2XA2,7XA6,1X8A6,6X1P2E11.3,I7)                   GEPLOT
      FLD(6,6,ITYTLS(1,I)) = FLD(30,6,ITYTLS(1,I))                       GEPLOT
      ITYTLS(1,I) = OR(AND(ITYTLS(1,I),O1),O2)                           GEPLOT
 1200 CONTINUE                                                           GEPLOT
C                                                               GEPLT453 GEPLOT
C                                                               GEPLT454 GEPLOT
C                                                START THE PLOTTINGGEPLT455 GEPLOT
 1220 WRITE (6,1240)                                            GEPLT456 GEPLOT
 1240 FORMAT(1H1,58X,14HSTARTING PLOTS/)                        GEPLT457 GEPLOT
C                               COMPUTE THE NUMBER OF GRIDS REQUIREDGEPLT458 GEPLOT
      NGRDS = .9999 + (TZ-TA)/TPG                               GEPLT459 GEPLOT
C                                  INITIALIZE THE ABSCISSA LIMITSGEPLT460 GEPLOT
      IR = 1                                                    GEPLT461 GEPLOT
      ABSR = TA                                                 GEPLT462 GEPLOT
C                                            CENTER THE CASE TITLEGEPLT463 GEPLOT
      NCA = NBLANK (TITLEA,12)                                  GEPLT464 GEPLOT
      NPA = 590 - 6*NCA                                         GEPLT465 GEPLOT
      NPX = 276                                                          GEPLOT
      NCB = 54                                                           GEPLOT
C                                         START THE GRID SET LOOPGEPLT466 GEPLOT
      DO 1620 I=1,NGRDS                                         GEPLT467 GEPLOT
      KFINIS = 0                                                GEPLT468 GEPLOT
C                                        SET THE LEFT-HAND LIMITSGEPLT469 GEPLOT
      IL = IR                                                   GEPLT470 GEPLOT
      ABSL = ABSR                                               GEPLT471 GEPLOT
```

```
      ABSR = ABSL+TPG                                           GEPLT472    GEPLOT
      ISW = 1                                                               GEPLOT
      NAVG = 0                                                              GEPLOT
      CALL DXDYV(1,ABSL,ABSR,DELX,NEX,LABX,NUMX,30.0,IERX)                  GEPLOT
      ZBSL = INT((ABSL-DELX)/DELX)*DELX                                     GEPLOT
      ZBSR = INT((ABSR+DELX)/DELX)*DELX                                     GEPLOT
      IF(ZBSL+DELX .LE. ABSL) ZBSL = ZBSL + DELX                            GEPLOT
      IF(ZBSR-DELX .GE. ABSR) ZBSR = ZBSR - DELX                            GEPLOT
      IF(ABSR .GE. 10.) NUMX=NUMX-1                                         GEPLOT
C                                          LOAD THE ABSCISSA VALUESGEPLT473 GEPLOT
C                                             SET THE ITEM COUNTERGEPLT484  GEPLOT
      J = 1                                                     GEPLT485    GEPLOT
C                                            SET THE CURVE COUNTERGEPLT486  GEPLOT
 1300 K = 1                                                     GEPLT487    GEPLOT
C                                         CLEAR THE SUBTITLE ARRAYSGEPLT488 GEPLOT
      DO 1320 L=1,20                                            GEPLT489    GEPLOT
      TITLEB(L) = BLANK                                         GEPLT490    GEPLOT
 1320 TITLEC(L) = BLANK                                         GEPLT491    GEPLOT
      CALL FILMAV(0)                                            GEPLT492    GEPLOT
      JC = J                                                    GEPLT494    GEPLOT
 1340 DO 1360 L=1,9                                             GEPLT495    GEPLOT
 1360 TITLEB(L) = TITLES(L,JC)                                  GEPLT496    GEPLOT
      NC = FLD(6,6,ITYTLS(1,JC)) - 48                                       GEPLOT
      IF (NC.GT.4) GO TO 1380  * TOO MANY CURVES                GEPLT498    GEPLOT
      IF (LOC(JC+1).LT.0) GO TO 1380  * ERROR ITEM              GEPLT499    GEPLOT
      FLD(0,30,TITLEB(1)) = FLD(0,30,PSYM(NC))                              GEPLOT
      NPY = 1005 - NC*18                                        GEPLT503    GEPLOT
      NMAR = NC                                                             GEPLOT
C                                             WRITE THE SUBTITLESGEPLT504   GEPLOT
      CALL RITE2V(NPX,NPY,1023,90,1,NCB,1,TITLEB,NL)            GEPLT505    GEPLOT
 1380 JC = JC + 1                                               GEPLT506    GEPLOT
      IF (ITEM(JC).GT.0) GO TO 1340                             GEPLT507    GEPLOT
      NPY = 1024 -(NPY-9)                                       GEPLT508    GEPLOT
      NMAR = (NMAR + 1)*18                                                  GEPLOT
      IF(NOAVG .GT. 0) NMAR = NMAR + 18                                     GEPLOT
C                                                   DRAW THE GRIDGEPLT509   GEPLOT
C        SUBROUTINE DXDYV CALCULATES CERTAIN ARGUMENTS FOR GRIDIV, SUCH AS  GEPLOT
C        THE INCREMENTS FOR LINE SPACING DELX AND DELY.  THE FOLLOWING      GEPLOT
C        PROCEDURE ADJUSTS THE  MAX AND MIN LIMITS OF THE GRID TO           GEPLOT
C        INSURE THAT THEY ARE INTEGRAL MULTIPLES OF THE INCREMENTS          GEPLOT
      CALL DXDYV(2,YLO(J),YHI(J),DELY,NEY,LABY,NUMY,33.0,IERY)              GEPLOT
      YLOJ = INT((YLO(J) - DELY)/DELY)*DELY                                 GEPLOT
      YHIJ = INT((YHI(J) + DELY)/DELY)*DELY                                 GEPLOT
      IF(YLOJ + DELY .LE. YLO(J)) YLDJ = YLOJ + DELY                        GEPLOT
      IF(YHIJ - DELY .GE. YHI(J)) YHIJ = YHIJ - DELY                        GEPLOT
      YLO(J) = YLOJ                                                         GEPLOT
      YHI(J) = YHIJ                                                         GEPLOT
      CALL SETMIV(140,20,50,NMAR)                                           GEPLOT
      CALL SETCIV(12,18)                                                    GEPLOT
      IF(LABY.EQ.10) LABY = 5                                               GEPLOT
      LABY = -LABY                                                          GEPLOT
      CALL GRIDIV(2,ZBSL,ZBSR,YLO(J),YHI(J),DELX,DELY,NEX,NEY,LABX,         GEPLOT
     1      LABY,NUMX,NUMY)                                                 GEPLOT
C                                                 LABEL THE AXESGEPLT511    GEPLOT
      L = IGS(J)                                                GEPLT512    GEPLOT
      CALL RITE2V(456,9,1023,90,1,24,1,BCDX,NL)                 GEPLT513    GEPLOT
      CALL RITE2V(92,380,1023,180,1,24,1,BCDY(1,L),NL)          GEPLT514    GEPLOT
C                                            WRITE THE CASE TITLEGEPLT515   GEPLOT
      CALL RITE2V(NPA,1005,1023,90,1,NCA,1,TITLEA,NL)           GEPLT516    GEPLOT
C                                       CHECK FOR TOO MANY CURVESGEPLT517   GEPLOT
 1400 IF(K.GT.4) GO TO 1440                                                 GEPLOT
C                                           CHECK FOR ERROR ITEMGEPLT519    GEPLOT
      IF(LOC(J+1).LT.0) GO TO 1480                                          GEPLOT
```

```
C                                   LOAD THE RESPECTIVE ORDINATE VALUESGEPLT521     GEPLOT
      IF(ISW .EQ. 0) GO TO 1409                                                      GEPLOT
      JJ = 0                                                                         GEPLOT
      DO 1403 KK=IL,NTYMS                                                            GEPLOT
      L = ITEMS*(KK-1) + 1                                                           GEPLOT
      IR=KK                                                                          GEPLOT
      IF(XY(L) .GT. ABSR) GO TO 1406                                                 GEPLOT
      JJ = JJ + 1                                                                    GEPLOT
 1403 ABS(JJ) = XY(L)                                                                GEPLOT
 1406 NPTS = JJ                                                                      GEPLOT
      ISW = 0                                                                        GEPLOT
 1409 DO 1420 L=1,NPTS                                                               GEPLOT
      M = ITEMS*(IL+L-2)+J+1                                            GEPLT523     GEPLOT
 1420 ORD(L) = XY(M)                                                    GEPLT524     GEPLOT
C                        BRANCH TO THE APPROPRIATE PLOTIV SUBROUTINE CALLGEPLT525     GEPLOT
      CALL GOPLOT(K)                                                    GEPLT526     GEPLOT
      GO TO 1520                                                                     GEPLOT
C                                                       TOO MANY CURVESGEPLT528     GEPLOT
 1440 WRITE (6,1460) (TITLES(L,J),L=1,9)                                GEPLT529     GEPLOT
 1460 FORMAT(15X,8HSKIPPING,5X,9A6,5X,28HTOO MANY CURVES ON THIS GRID)  GEPLT530     GEPLOT
      GO TO 1550                                                                     GEPLOT
C                                                            ERROR ITEMGEPLT532     GEPLOT
 1480 WRITE (6,1500) (TITLES(L,J),L=1,9)                                GEPLT533     GEPLOT
 1500 FORMAT(15X,8HSKIPPING,5X,9A6,5X,21HTHIS ITEM IS IN ERROR)         GEPLT534     GEPLOT
      GO TO 1550                                                                     GEPLOT
 1520 WRITE (6,1540) (TITLES(L,J),L=1,9)                                GEPLT536     GEPLOT
 1540 FORMAT(15X,8HPLOTTING,5X,9A6)                                     GEPLT537     GEPLOT
C                                     BUMP THE ITEM AND CURVE COUNTERSGEPLT538     GEPLOT
 1550 IF(NOAVG .EQ. 0) GO TO 1575                                                    GEPLOT
      IF(J .NE. IABS(ITMAVG(NAVG+1))) GO TO 1575                                     GEPLOT
      NAVG = NAVG + 1                                                                GEPLOT
      IF(NOAVG .EQ. NAVG) NOAVG = 0                                                  GEPLOT
      IF(ITMAVG(NAVG) .GT. 0) GO TO 1575                                             GEPLOT
      IF(K+1.GT. 4) GO TO 1575                                                       GEPLOT
      IF(LOC(J+1) .LT. 0) GO TO 1575                                                 GEPLOT
      NC = FLD(6,6,TTYTLS(1,J)) - 47                                                 GEPLOT
      K = K + 1                                                                      GEPLOT
      TITLEB(1) = PSYM(K)                                                            GEPLOT
      TITLEB(2) = 6HAVERAG                                                           GEPLOT
      TITLEB(3) = 6HE                                                                GEPLOT
      DO 1553 L=4,20                                                                 GEPLOT
 1553 TITLEB(L) = 6H                                                                 GEPLOT
      NPY =1005 - 18*NC                                                              GEPLOT
      CALL RITE2V(NPX,NPY,1023,90,1,NCB,1,TITLEB,NL)                                 GEPLOT
      ISW = 1                                                                        GEPLOT
      DO 1556 L=1,NTPTS                                                              GEPLOT
      ABS(L) = ABS(ISTART+L)                                                         GEPLOT
 1556 ORD(L) = AVG(NAVG)                                                             GEPLOT
      NPTS = NTPTS                                                                   GEPLOT
      CALL GOPLOT(K)                                                                 GEPLOT
 1560 CONTINUE                                                                       GEPLOT
 1575 J = J + 1                                                                      GEPLOT
      K = K+1                                                           GEPLT540     GEPLOT
C                                              CHECK FOR END OF ITEMSGEPLT541     GEPLOT
      IF (J .LT. ITEMS) GO TO 1580                                      GEPLT542     GEPLOT
      KFINIS = 1                                                        GEPLT543     GEPLOT
      GO TO 1600                                                        GEPLT544     GEPLOT
C                                                  CHECK FOR NEW GRIDGEPLT545     GEPLOT
 1580 IF (ITEM(J) .GT. 0) GO TO 1400                                    GEPLT546     GEPLOT
 1600 IF (KFINIS.EQ.0) GO TO 1300                                       GEPLT547     GEPLOT
C                                        REFERENCE THE NEW GRID SETSGEPLT548     GEPLOT
      IL = IR                                                           GEPLT549     GEPLOT
      ABSL = ABSR                                                       GEPLT550     GEPLOT
```

```
 1620 CONTINUE                                              GEPLT551       GEPLOT
      IBB = 0                                               GEPLT552       GEPLOT
 1640 CALL FILMAV(0)                                        GEPLT553       GEPLOT
      NCASE = NCASE + 1                                     GEPLT554       GEPLOT
      GO TO 240                                             GEPLT555       GEPLOT
      END                                                   GEPLT556       GEPLOT
```

GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT GOPLOT

```
      SUBROUTINE  GOPLOT(NSYM)                                           GOPLOT
      DIMENSION  ASYM(50),OSYM(50),BUFA(4000),ABS(1),ORD(1)              GOPLOT
      COMMON  NPTS,TPG,BUFA                                              GOPLOT
      EQUIVALENCE (BUFA(1),ABS(1)),(BUFA(2001),ORD(1))                   GOPLOT
      INTEGER  ISYM(2)                                                   GOPLOT
      DATA  ISYM/6H1234  ,6H      /                                      GOPLOT
      FNP = NSYM*2 + 6                                                   GOPLOT
      DT = ABS(NPTS) - ABS(1)                                            GOPLOT
      NP = FNP + DT/TPG + 1.5                                            GOPLOT
      NPP = NP - 1                                                       GOPLOT
      K = NPTS / NPP                                                     GOPLOT
      ASYM(1) = ABS(1)                                                   GOPLOT
      OSYM(1) = ORD(1)                                                   GOPLOT
      ASYM(NP) = ABS(NPTS)                                               GOPLOT
      OSYM(NP) = ORD(NPTS)                                               GOPLOT
      KK = 0                                                             GOPLOT
      DO 10 I=2,NPP                                                      GOPLOT
      KK = KK + K                                                        GOPLOT
      IX=NXV(ABS(KK))                                                    GOPLOT
      IY=NYV(ORD(KK))                                                    GOPLOT
      CALL RITE2V(IX,IY,1023,90,1,1,NSYM,ISYM,NL)                        GOPLOT
   10 CONTINUE                                                           GOPLOT
      IXA = NXV(ABS(1))                                                  GOPLOT
      IYA=NYV(ORD(1))                                                    GOPLOT
      DO 15 I=2,NPTS                                                     GOPLOT
      IXB = NXV(ABS(I))                                                  GOPLOT
      IYB =NYV(ORD(I))                                                   GOPLOT
      CALL LINEV(IXA,IYA,IXB,IYB)                                        GOPLOT
      IXA=IXB                                                            GOPLOT
      IYA = IYB                                                          GOPLOT
   15 CONTINUE                                                           GOPLOT
      RETURN                                                             GOPLOT
      END
```

```
      SUBROUTINE HEATER(Q,QHT,KODE,TSEN,TON,TOFF)                 HEATER
C                                                                 HEATER
      IF(TSEN .LT. TON ) GO TO 200                                HEATER
      IF(TSEN .GT. TOFF) GO TO 100                                HEATER
      Q = Q + QHT*KODE                                            HEATER
      RETURN                                                      HEATER
  100 KODE = 0                                             *NEW   HEATER
      RETURN                                               **-1   HEATER
  200 KODE = 1                                                    HEATER
      Q = Q + QHT                                                 HEATER
      RETURN                                                      HEATER
      END                                                         HEATER
```

```fortran
      SUBROUTINE HSTRY(PR,VP,W,TINC)
C
      DIMENSION PR(1), VP(1), W(1)
C
      COMMON /TEMP  / T(1)
      COMMON /POINTN/ LNODE
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)
      COMMON /FIXCON/ CON(1)
      COMMON /TITLE / HEADER(12)
      COMMON /DIMENS/ NNA, NND, NSL
C
      EQUIVALENCE (NPR,PRN), (NVP,VPN), (NW,WN)
C
      DATA KK / 0/
      DATA LL / 0/
      DATA LT /23/
C
C
      DT  = CON(1)
      PRN = PR (1)
      VPN = VP (1)
      WN  = W  (1)
      IF(KK .GT. 0) GO TO 10
      IF(NPR .LT. 1) NPR = 1
      IF(NVP .LT. 1) NVP = 1
      IF(NW  .LT. 1) NW  = 1
      IF(LNODE .EQ. 0) CALL NNREAD(1)
      WRITE(LT) HEADER, (LL,I=1,6), NPR, NVP, LL,LL,LL, NW, LL,LL, NSL,
     1 (NEXT(LNODE+I),I=1,NSL)
      TIME2 = 0.0
      TIME1 = CON(13) + CON(2)
      CALL HSTTP(TIME1)
      KK = 1
      GO TO 50
   10 TIME2 = TIME2 + DT
      IF(CON(1)+1.000001 .LT. CON(3)) GO TO 12
      GO TO 15
   12 IF(TIME2  .LT. TINC ) GO TO 50
      IF(CON(1) .LT. TIME1) GO TO 50
   15 TIME1 = CON(1)
      TIME2 = 0.0
      CALL HSTTP(CON(1))
      IF(CON(1)+1.000001 .LT. CON(3)) GO TO 50
   20 CALL HSTTP(-CON(1))
      KK = 0
      END FILE LT
   50 CONTINUE
      RETURN
C
C
      SUBROUTINE HSTTP(XTIME)
C
      WRITE(LT) XTIME, (PR(I+1),I=1,NPR), (VP(I+1),I=1,NVP),
     1                  (W (I+1),I=1,NW ), (T (I   ),I=1,NSL)
      RETURN
      END
```

HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT HXCNT

```
      SUBROUTINE HXCNT (X1,X2,X3,X4,X5,X6,X7,X8,X9)                            HXCNT
C  ANAYSIS OF COUNTER FLOW HEAT EXCHANGERS                                     HXCNT
      DIMENSION CP(2), FR(2), NCP(2), TIN(2), TOUT(2), WCP(2)                  HXCNT
      EQUIVALENCE (NUA,UA), (NCP,CP)                                           HXCNT
C                                                                              HXCNT
      UA = X1                                                                  HXCNT
      FR(1) = X2                                                               HXCNT
      FR(2) = X3                                                               HXCNT
      CP(1) = X4                                                               HXCNT
      CP(2) = X5                                                               HXCNT
      TIN(1) = X6                                                              HXCNT
      TIN(2) = X7                                                              HXCNT
      TOUT(1) = X8                                                             HXCNT
      TOUT(2) = X9                                                             HXCNT
      DO 10 I=1,2                                                              HXCNT
      IF(FR(I) .LT. 0.0) GO TO 100                                             HXCNT
   10 CONTINUE                                                                 HXCNT
      IF(NCP(1) .LT. 1 .OR. NCP(1) .GT. 65000) GO TO 3                         HXCNT
      TAVG = 0.5*(TIN(1)+TOUT(1))                                              HXCNT
      CALL D1DEGI(TAVG,X4,CP(1))                                               HXCNT
    3 IF(NCP(2) .LT. 1 .OR. NCP(2) .GT. 65000) GO TO 6                         HXCNT
      TAVG = 0.5*(TIN(2)-TOUT(2))                                              HXCNT
      CALL D1DEGI(TAVG,X5,CP(2))                                               HXCNT
    6 CONTINUE                                                                 HXCNT
      WCP(1) = FR(1)*CP(1)                                              *NEW   HXCNT
      WCP(2) = FR(2)*CP(2)                                              *NEW   HXCNT
      IF(IABS(NUA) .LE. 99999 .AND. IABS(NUA) .GT. 0)                          HXCNT
     X CALL D2DEGI(FR(1),FR(2),X1,UA)                                          HXCNT
      IS = 1                                                                   HXCNT
      IL = 2                                                                   HXCNT
      IF(WCP(1) .LE. WCP(2)) GO TO 20                                          HXCNT
      IS = 2                                                                   HXCNT
      IL = 1                                                                   HXCNT
   20 WCPRAT = WCP(IS)/WCP(IL)                                                 HXCNT
      IF(WCPRAT .GT. .001) GO TO 30                                            HXCNT
      EFF = 1.0                                                                HXCNT
      GO TO 50                                                                 HXCNT
   30 IF(WCPRAT .LT. .999 .OR. WCPRAT .GT. 1.001) GO TO 40                     HXCNT
      EFF = UA/(WCP(IS)+UA)                                                    HXCNT
      GO TO 50                                                                 HXCNT
   40 E = EXP(-UA/WCP(IS) + UA/WCP(IL))                                        HXCNT
      EFF = (1.-E)/(1.-WCPRAT*E)                                               HXCNT
   50 TOUT(IS) = TIN(IS) - EFF*(TIN(IS)-TIN(IL))                               HXCNT
      TOUT(IL) = TIN(IL) + WCPRAT*(TIN(IS)-TOUT(IS))                           HXCNT
      X8 = TOUT(1)                                                             HXCNT
      X9 = TOUT(2)                                                             HXCNT
      RETURN                                                                   HXCNT
  100 WRITE(6,101) FR(I)                                                       HXCNT
  101 FORMAT(1H0 131(1H*)/// THE NEGATIVE FLOW RATE'E15.8,' IS NOT ALLOW       HXCNT
     XED. EXECUTION TERMINATED IN SUBROUTINE HXCNT'//1X 131(1H*))              HXCNT
      CALL WLKBCK                                                              HXCNT
      CALL EXIT                                                                HXCNT
      END                                                                      HXCNT
```

HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS HXCROS

```
      SUBROUTINE HXCROS(X1,X2,X3,X4,X5,X6,X7,X8,X9,KODE)                    HXCROS
C  ANALYSIS OF CROSS FLOW HEAT EXCHANGERS                                   HXCROS
      DIMENSION CP(2), FR(2), NCP(2), TIN(2), TOUT(2), WCP(2)               HXCROS
      EQUIVALENCE (NUA,UA), (NCP,CP)                                        HXCROS
C                                                                           HXCROS
      UA = X1                                                               HXCROS
      FR(1) = X2                                                            HXCROS
      FR(2) = X3                                                            HXCROS
      CP(1) = X4                                                            HXCROS
      CP(2) = X5                                                            HXCROS
      TIN(1) = X6                                                           HXCROS
      TIN(2) = X7                                                           HXCROS
      TOUT(1) = X8                                                          HXCROS
      TOUT(2) = X9                                                          HXCROS
      DO 10 I=1,2                                                           HXCROS
      IF(FR(I) .LT. 0.0) GO TO 100                                          HXCROS
   10 CONTINUE                                                              HXCROS
      IF(NCP(1) .LT. 1 .OR. NCP(1) .GT. 65000) GO TO 3                      HXCROS
      TAVG = 0.5*(TIN(1)+TOUT(1))                                           HXCROS
      CALL D1DEG1(TAVG,X4,CP(1))                                            HXCROS
    3 IF(NCP(2) .LT. 1 .OR. NCP(2) .GT. 65000) GO TO 6                      HXCROS
      TAVG = 0.5*(TIN(2)-TOUT(2))                                           HXCROS
      CALL D1DEG1(TAVG,X5,CP(2))                                            HXCROS
    6 CONTINUE                                                              HXCROS
      WCP(1) = FR(1)*CP(1)                                        *NEW      HXCROS
      WCP(2) = FR(2)*CP(2)                                        *NEW      HXCROS
      IF(IABS(NUA) .LE. 99999 .AND. IABS(NUA) .GT. 0)                       HXCROS
     X CALL D2DEG1(FR(1),FR(2),X1,UA)                                       HXCROS
      IS = 1                                                                HXCROS
      IL = 2                                                                HXCROS
      IF(WCP(1) .LE. WCP(2)) GO TO 20                                       HXCROS
      IS = 2                                                                HXCROS
      IL = 1                                                                HXCROS
   20 WCPRAT = WCP(IS)/WCP(IL)                                              HXCROS
      IF(WCPRAT .GT. .001) GO TO (30,40,50,60), KODE                        HXCROS
      EFF = 1.0                                                             HXCROS
      GO TO 70                                                              HXCROS
   30 E = EXP(-UA**.78*WCP(IS)**.22/WCP(IL)) - 1.                           HXCROS
      EFF = 1. - EXP(E*WCP(IL)*UA**.22/WCP(IS)**1.22)                       HXCROS
      GO TO 70                                                              HXCROS
   40 UAS = UA/WCP(IS)                                                      HXCROS
      UAL = UA/WCP(IL)                                                      HXCROS
      EFF = UAS/(UAS/(1.-EXP(-UAS)) + UAL/(1.-EXP(-UAL)) - 1.)              HXCROS
      GO TO 70                                                              HXCROS
   50 EFF = (1.-EXP(-WCPRAT)*(1.-EXP(-UA/WCP(IS))))/WCPRAT                  HXCROS
      GO TO 70                                                              HXCROS
   60 EFF = 1. - EXP(-WCP(IL)/WCP(IS))*(1.-EXP(-UA/WCP(IL)))                HXCROS
   70 TOUT(IS) = TIN(IS) - EFF*(TIN(IS)-TIN(IL))                            HXCROS
      TOUT(IL) = TIN(IL) + WCPRAT*(TIN(IS)-TOUT(IS))                        HXCROS
      X8 = TOUT(1)                                                          HXCROS
      X9 = TOUT(2)                                                          HXCROS
      RETURN                                                                HXCROS
  100 WRITE(6,101) FR(I)                                                    HXCROS
  101 FORMAT(1H0 131(1H*)//' THE NEGATIVE FLOW RATE'E15.8,' IS NOT ALLOW    HXCROS
     XED. EXECUTION TERMINATED IN SUBROUTINE HXCROS'//1X 131(1H*))          HXCROS
      CALL WLKBCK                                                           HXCROS
      CALL EXIT                                                             HXCROS
      END                                                                   HXCROS
```

```
      SUBROUTINE HXEFF (X1,X2,X3,X4,X5,X6,X7,X8,X9)                         HXEFF
C  ANALYSIS OF HEAT EXCHANGERS WITH EFFECTIVENESS GIVEN                     HXEFF
      DIMENSION CP(2), FR(2), NCP(2), TIN(2), TOUT(2), WCP(2)               HXEFF
      EQUIVALENCE (NEFF,EFF), (NCP,CP)                                      HXEFF
C                                                                           HXEFF
      EFF = X1                                                              HXEFF
      FR(1) = X2                                                            HXEFF
      FR(2) = X3                                                            HXEFF
      CP(1) = X4                                                            HXEFF
      CP(2) = X5                                                            HXEFF
      TIN(1) = X6                                                           HXEFF
      TIN(2) = X7                                                           HXEFF
      TOUT(1) = X8                                                          HXEFF
      TOUT(2) = X9                                                          HXEFF
      DO 10 I=1,2                                                           HXEFF
      IF(FR(I) .LT. 0.0) GO TO 100                                          HXEFF
   10 CONTINUE                                                              HXEFF
      IF(NCP(1) .LT. 1 .OR. NCP(1) .GT. 65000) GO TO 3                      HXEFF
      TAVG = 0.5*(TIN(1)+TOUT(1))                                           HXEFF
      CALL D1DEG1(TAVG,X4,CP(1))                                            HXEFF
    3 IF(NCP(2) .LT. 1 .OR. NCP(2) .GT. 65000) GO TO 6                      HXEFF
      TAVG = 0.5*(TIN(2)-TOUT(2))                                           HXEFF
      CALL D1DEG1(TAVG,X5,CP(2))                                            HXEFF
    6 CONTINUE                                                              HXEFF
      WCP(1) = FR(1)*CP(1)                                        *NEW      HXEFF
      WCP(2) = FR(2)*CP(2)                                        *NEW      HXEFF
      IF(IABS(NEFF) .LE. 99999 .AND. IABS(NEFF) .GT. 0)                     HXEFF
     X CALL D2DEG1(FR(1),FR(2),X1,EFF)                            *NEW      HXEFF
      IS = 1                                                      **-1      HXEFF
      IL = 2                                                                HXEFF
      IF(WCP(1) .LE. WCP(2)) GO TO 20                                       HXEFF
      IS = 2                                                                HXEFF
      IL = 1                                                                HXEFF
   20 TOUT(IS) = TIN(IS) - EFF*(TIN(IS)-TIN(IL))                            HXEFF
      TOUT(IL) = TIN(IL) + WCP(IS)/WCP(IL)*(TIN(IS)-TOUT(IS))               HXEFF
      X8 = TOUT(1)                                                          HXEFF
      X9 = TOUT(2)                                                          HXEFF
      RETURN                                                                HXEFF
  100 WRITE(6,101) FR(I)                                                    HXEFF
  101 FORMAT(1H0 131(1H*)//' THE NEGATIVE FLOW RATE'E15.8,' IS NOT ALLOW    HXEFF
     XED. EXECUTION TERMINATED IN SUBROUTINE HXEFF'//1X 131(1H*))           HXEFF
      CALL WLKBCK                                                           HXEFF
      CALL EXIT                                                             HXEFF
      END                                                                   HXEFF
```

```
      SUBROUTINE HXPAR (X1,X2,X3,X4,X5,X6,X7,X8,X9)                     HXPAR
C  ANALYSIS OF PARALLEL FLOW HEAT EXCHANGERS                            HXPAR
      DIMENSION CP(2), FR(2), NCP(2), TIN(2), TOUT(2), WCP(2)           HXPAR
      EQUIVALENCE (NUA,UA), (NCP,CP)                                    HXPAR
C                                                                       HXPAR
      UA = X1                                                           HXPAR
      FR(1) = X2                                                        HXPAR
      FR(2) = X3                                                        HXPAR
      CP(1) = X4                                                        HXPAR
      CP(2) = X5                                                        HXPAR
      TIN(1) = X6                                                       HXPAR
      TIN(2) = X7                                                       HXPAR
      TOUT(1) = X8                                                      HXPAR
      TOUT(2) = X9                                                      HXPAR
      DO 10 I=1,2                                                       HXPAR
      IF(FR(I) .LT. 0.0) GO TO 100                                      HXPAR
   10 CONTINUE                                                          HXPAR
      IF(NCP(1) .LT. 1 .OR. NCP(1) .GT. 65000) GO TO 3                  HXPAR
      TAVG = 0.5*(TIN(1)+TOUT(1))                                       HXPAR
      CALL D1DEG1(TAVG,X4,CP(1))                                        HXPAR
    3 IF(NCP(2) .LT. 1 .OR. NCP(2) .GT. 65000) GO TO 6                  HXPAR
      TAVG = 0.5*(TIN(2)-TOUT(2))                                       HXPAR
      CALL D1DEG1(TAVG,X5,CP(2))                                        HXPAR
    6 CONTINUE                                                          HXPAR
      WCP(1) = FR(1)*CP(1)                                    *NEW      HXPAR
      WCP(2) = FR(2)*CP(2)                                    *NEW      HXPAR
      IF(IABS(NUA) .LE. 99999 .AND. IABS(NUA) .GT. 0)                   HXPAR
     X CALL D2DEG1(FR(1),FR(2),X1,UA)                                   HXPAR
      IS = 1                                                            HXPAR
      IL = 2                                                            HXPAR
      IF(WCP(1) .LE  WCP(2)) GO TO 20                                   HXPAR
      IS = 2                                                            HXPAR
      IL = 1                                                            HXPAR
   20 WCPRAT = WCP(IS)/WCP(IL)                                          HXPAR
      IF(WCPRAT .GT. .001) GO TO 30                                     HXPAR
      EFF = 1.0                                                         HXPAR
      GO TO 50                                                          HXPAR
   30 EFF = (1.-EXP(-UA/WCP(IS)-UA/WCP(IL)))/(1.+WCPRAT)                HXPAR
   50 TOUT(IS) = TIN(IS) - EFF*(TIN(IS)-TIN(IL))                        HXPAR
      TOUT(IL) = TIN(IL) + WCPRAT*(TIN(IS)-TOUT(IS))                    HXPAR
      X8 = TOUT(1)                                                      HXPAR
      X9 = TOUT(2)                                                      HXPAR
      RETURN                                                            HXPAR
  100 WRITE(6,101) FR(I)                                                HXPAR
  101 FORMAT(1H0 131(1H*)//' THE NEGATIVE FLOW RATE'E15.8,' IS NOT ALLOW HXPAR
     XED. EXECUTION TERMINATED IN SUBROUTINE HXPAR'//1X 131(1H*))       HXPAR
      CALL WLKBCK                                                       HXPAR
      CALL EXIT                                                         HXPAR
      END                                                               HXPAR
```

LINECK LINECK LINECK LINECK LINECK LINECK LINECK LINECK LINECK LINECK LINECK

```
      SUBROUTINE LINECK(I)                                              LINECK
C                                                                       LINECK
      COMMON /FIXCON/ N(1)                                              LINECK
C                                                                       LINECK
C                                                                       LINECK
      IF(N(28)+I .GT. 60 .OR. N(29) .EQ. 0) CALL TOPLIN                 LINECK
      N(28) = N(28) + I                                                 LINECK
      RETURN                                                            LINECK
      END                                                               LINECK
                                                                        LINECK
```

```
      SUBROUTINE LINRV (I,LREFA,IBASE,ITOP,XYL,XYU,DXY,NM,IJ,NXY,IWH)      LINRV
      DIMENSION IXY1(2),IXY2(2) ,XCOUNT(3)                                 LINRV
C     DIMENSION IXY1(2),IXY2(2) ,XCOUNT(3)                                 LINRV
      EQUIVALENCE(XYN,T),(XYNN,ADELT), (DELNI,XY) ,(TT,NDMX)               LINRV
         IJK       = 1                                          MSC- AA    LINRV
           CALL   LGST(  ISTAY, MI, MA  )                       PL- 0001   LINRV
      XCOUNT(2)= IABS(NM)                                                  LINRV
      XCOUNT(3)=IABS(IJ)                                                   LINRV
      DELTA=ABS(DXY)                                                       LINRV
      CALL BNBCDV(XYL,BBB,NDD)                                             LINRV
      CALL BNBCDV(XYU,BBC,NDE)                                             LINRV
      IF(NDD.GE.NDE) GO TO 60                                              LINRV
      NDT = NDE                                                            LINRV
      GO TO 70                                                             LINRV
   60 NDT =NDD                                                             LINRV
   70 CONTINUE                                                             LINRV
         IF( -DELTA )  80, 3740, 80                                        LINRV
   80 IXY1(1)=IBASE                                                        LINRV
      IXY2(1)=ITOP                                                         LINRV
      IX=LREFA                                                             LINRV
      IXY1(2)=IXY1(1)                                                      LINRV
      IXY2(2)=IXY2(1)                                                      LINRV
      IY=IX                                                                LINRV
      IWHO2=IWH/2                                                          LINRV
      STOP1=AMAX1(XYU,XYL)                                                 LINRV
      STOP2=AMIN1(XYU,XYL)                                                 LINRV
      T= AMAX1(ABS(STOP1),ABS(STOP2))                                      LINRV
      CALL BNBCDV(T,XY,NDMAX)                                              LINRV
      NDMAX=MAX0(0,NDMAX)                                                  LINRV
      TRIAL=0.0                                                            LINRV
      IF (STOP1*STOP2) 170,170,110                                         LINRV
  110 TRIAL=STOP2                                                          LINRV
      IF (STOP1) 140,170,170                                               LINRV
  140 TRIAL=STOP1                                                          LINRV
  170 XCOUNT(1)=1.0                                                        LINRV
      TREF=0.0                                                             LINRV
      TT=AMAX1(XCOUNT(2),1.0)*AMAX1(XCOUNT(3),1.0)                         LINRV
      DELNI=DELTA*TT                                                       LINRV
      ADELT=9999999.0*DELTA                                                LINRV
      IF (ADELT-ABS(TRIAL)) 230,320,320                                    LINRV
  230 T=AINT(TRIAL/DELNI) *TT                                              LINRV
      IF (T) 260,290,260                                                   LINRV
  260 TREF= T*DELTA                                                        LINRV
      IF (ADELT-ABS(TRIAL-TREF)) 290,320,320                               LINRV
  290 TREF=TRIAL                                                           LINRV
  320 DO 1160 K=1,3                                                        LINRV
         DELTA      = ABS( DELTA )                              MSC-0001   LINRV
      IF (XCOUNT(K)) 380,1160,380                                          LINRV
  380 T=DELTA* XCOUNT(K)                                                   LINRV
      XYNN= AINT(ABS(TRIAL-TREF)/T) *XCOUNT(K)                             LINRV
      DO 1130 J=1,2                                                        LINRV
      XYN=XYNN                                                             LINRV
  470 XY= TREF+ DELTA*XYN                                                  LINRV
      GO TO (530,560),I                                                    LINRV
  530 IXY1(1)=NXV(XY)                                                      LINRV
      GO TO 590                                                            LINRV
  560 IXY1(2)=NYV(XY)                                                      LINRV
  590 IXY2(I)=IXY1(I)                                                      LINRV
      IF (IXY1(I)) 620,620,710                                             LINRV
```

```
      620    CONTINUE                                          MSC- 00U     LINRV
          IF(XY.GT.STOP1.AND.DELTA.GT.0.0) GO TO 1100                       LINRV
          IF(XY.LT.STOP2.AND.DELTA.LT.0.0) GO TO 1100                       LINRV
          IF(XY.EQ.STOP1.OR.XY.EQ.STOP2) GO TO 710                          LINRV
          GO TO 1075                                                        LINRV
 710      CONTINUE                                                          LINRV
          GO  TO    ( 3504,  1160,  770  ),  K                 MSC- AA      LINRV
 3504     CONTINUE                                             MSC- AA      LINRV
          GO  TO    ( 3508, 3512, 3520 ),  IJK                 MSC- AA      LINRV
 3508     CONTINUE                                             MSC- AA      LINRV
          IF(  IABS( NM )  .LE.  1  )   GO  TO    3516         MSC- AA      LINRV
          JTM        = IABS( NM ) - 1                          MSC- AA      LINRV
          IJK        = 2                                       MSC- AA      LINRV
 3512     CONTINUE                                             MSC- AA      LINRV
          JTM        = JTM + 1                                 MSC- AA      LINRV
          IF( JTM  .LT.  IABS( NM )  )  GO  TO    3520         MSC- AA      LINRV
          JTM        = 0                                       MSC- AA      LINRV
          CALL       LWIDTH(  MA  )                            MSC- AA      LINRV
          GO  TO     3520                                      MSC- AA      LINRV
 3516     CONTINUE                                             MSC- AA      LINRV
          IJK        = 3                                       MSC- AA      LINRV
 3520     CONTINUE                                             MSC- AA      LINRV
          CALL       VLAGM(  IXY1( 1 ), IXY1( 2 ), IXY2( 1 ), IXY2( 2 ),I) MSC- AA      LINRV
          CALL       LWIDTH(  MI  )                            MSC- AA      LINRV
      GO TO 1070                                               MSC- AA      LINRV
  770 NDMX=NDMAX                                                            LINRV
      KNXY=NXY                                                              LINRV
      NCHAR=KNXY                                                            LINRV
      IF (KNXY) 800,1160,830                                                LINRV
  800 NCHAR = 4 - KNXY                                                      LINRV
      NDMAX=2                                                               LINRV
  830 IF (XY) 950,860,950                                                   LINRV
  860 NDMX=MIN0(NCHAR,NDMAX)                                                LINRV
      NCHAR=2                                                               LINRV
      IF (NDMAX) 890,920,890                                                LINRV
  890 NCHAR=NDMX                                                            LINRV
  920 KNXY=NCHAR                                                            LINRV
  950 GO TO (980,1010),I                                                    LINRV
  980 IX=IXY1(1)-(NCHAR*IWH)/2+IWHO2                                        LINRV
      GO TO 1040                                                            LINRV
 1010 IY=IXY1(2)+IWHO2-10                                                   LINRV
C                                                                           LINRV
C                                        BEGIN PRINTING LABELS              LINRV
C                                                                           LINRV
 1040 CALLB4BCDV(XY,BCDWD,NDS)                   CONVERT TO BCD             LINRV
      IXX = IX                                                              LINRV
      IMIN=IXX-12                                                           LINRV
      IF(NDS.LT.1) GO TO 1046                                               LINRV
      NNN=NDT-NDS                                                           LINRV
      IF(NNN.EQ.0) GO TO 1500                                               LINRV
C                                                                           LINRV
      IF(I.EQ.1) GO TO 1500                 SKIPPING LEADING SPACES         LINRV
      IXX = IX + NNN*12                                                     LINRV
      IMIN = IXX - 12                                                       LINRV
 1500 CONTINUE                                                              LINRV
C                                                                           LINRV
      CALL RITE2V(IXX,IY,3000,90,1,NDS,1,BCDWD,NLAST)  PRINT LEADING NUMERALS LINRV
      IF(XY.EQ.0) GO TO 1068                                                LINRV
      GXY = ABS(XY)                                                         LINRV
      INN = GXY + .5                                                        LINRV
      XXY = INN                                                             LINRV
      IF(ABS(XXY-GXY) .LT. .00001) GO TO 1065                               LINRV
      IXX =IXX + NDS *12                                                    LINRV
                                                                            LINRV
```

```
      C                                              PRINT DECIMAL POINT              LINRV
       1046 CALL RITE2V(IXX,IY,3000,90,1,1,1,1H.,NLAST)                               LINRV
            IXX=IXX+10                                                                LINRV
            NOC=1                                                                     LINRV
            IF(NDS.LE.0) GO TO 1048                                                   LINRV
            NOC = NDS+1                                                               LINRV
            GO TO 1049                                                                LINRV
       1048 IF(NDS.EQ.0) GO TO 1049                                                   LINRV
            NTT = ABS(NDS)                                                            LINRV
            DO 1047 IND=1,NTT                                                         LINRV
      C                                              WRITE ZEROS IN FRACTION          LINRV
            CALL RITE2V(IXX,IY,3000,90,1,1,1,1H0,NLAST)                               LINRV
            IXX=IXX+12                                                                LINRV
       1047 CONTINUE                                                                  LINRV
       1049 CONTINUE                                                                  LINRV
      C                                              PRINT TRAILING NUMERALS          LINRV
            CALL RITE2V(IXX,IY,3000,90,1,1,NOC,BCDWD,NLAST)                           LINRV
       1065 IF(XY.GT.0) GO TO 1068                                                    LINRV
      C                                              PRINT MINUS SIGN                 LINRV
            CALL RITE2V(IMIN,IY,3000,90,1,1,1,1H-,NLAST)                              LINRV
       1068 CONTINUE                                                                  LINRV
       1070 XYN=XYN+XCOUNT(K)                                                         LINRV
            GO TO 470                                                                 LINRV
       1100 DELTA=-DELTA                                                              LINRV
       1130 CONTINUE                                                                  LINRV
       1160 CONTINUE                                                                  LINRV
       3740    CONTINUE                                                               LINRV
               CALL       LWIDTH(   ISTAY   )                          MSC- AA        LINRV
               RETURN                                                                 LINRV
            END                                                                       LINRV
```

MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB MCOMB

```
      DIMENSION NBUFR(27), DATA(3000), ALPHA(15)                      MCOMB
      DIMENSION XSTART(7),XSTOP(7)                                    MCOMB
      DIMENSION ADD(7)                                                MCOMB
      DATA XSTART, XSTOP, ADD/21*0./                                  MCOMB
      DATA ALPHA /1HA,1HB,1HC,1HD,1H5,1H6,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,  MCOMB
     1           1HL,1HM/                                             MCOMB
      WRITE(6,3)                                                      MCOMB
    3 FORMAT(1H110X'OUTPUT FROM COMBIN ROUTINE'//)                    MCOMB
      READ(5,120) NTAPE, IUNIT, KT, KODE2, INC                        MCOMB
  120 FORMAT(6I5)                                                     MCOMB
      IF(NTAPE .EQ. 0) GO TO 200                                      MCOMB
      IF( KT .EQ. 0) KT = 13                                          MCOMB
      IF(IUNIT .EQ. 0) IUNIT = 7                                      MCOMB
      REWIND KT                                                       MCOMB
      KODE1 = 0                                                       MCOMB
      IF(NTAPE .LT. 0) KODE1 = 1                                      MCOMB
      NTAPE = IABS(NTAPE)                                             MCOMB
      IF(KODE1 .NE. 0) READ(5,27) (XSTART(I), XSTOP(I), I=1,NTAPE)    MCOMB
      IF(KODE2 .NE. 0) READ(5,140) ADD                                MCOMB
  140 FORMAT(7F10.0)                                                  MCOMB
   27 FORMAT(14F5.3)                                                  MCOMB
      DO 18 L = 1,NTAPE                                               MCOMB
      M=0                                                             MCOMB
      I = L + IUNIT - 1                                               MCOMB
      REWIND I                                                        MCOMB
      READ (I) NBUFR                                                  MCOMB
      IF (L .NE. 1) GO TO 8                                           MCOMB
      WRITE (KT) NBUFR                                                MCOMB
      NTOTAL = 0                                                      MCOMB
      DO 6 J=17,27                                                    MCOMB
      NTOTAL = NTOTAL + NBUFR(J)                                      MCOMB
    6 CONTINUE                                                        MCOMB
    9 READ (I) TIME,(DATA(K),K=1,NTOTAL)                              MCOMB
      TIME = TIME + ADD(L)                                            MCOMB
      IF (TIME .LT. 0.0 .AND. L .NE. NTAPE) GO TO 15                  MCOMB
      IF (TIME .LT. 0.0) GO TO 10                                     MCOMB
      IF(TIME-XSTART(L))9,                                            MCOMB
      IF(XSTOP(L))33,33                                               MCOMB
      IF(TIME-XSTOP(L))33,                                            MCOMB
      TIME=-TIME                                                      MCOMB
      IF(L-NTAPE)15,10,                                               MCOMB
   33 M=M-1                                                           MCOMB
      IF(M .GT. 0) GO TO 9                                            MCOMB
      M = INC                                                         MCOMB
      WTIME = TIME                                                    MCOMB
   10 WRITE(KT) TIME, (DATA(K),K=1,NTOTAL)                            MCOMB
      IF (TIME) 12,9,9                                                MCOMB
    8 READ (I) TIME,(DATA(K),K=1,NTOTAL)                              MCOMB
      TIME = TIME + ADD(L)                                            MCOMB
      IF(TIME-XSTART(L))8,                                            MCOMB
      IF(TIME-XTIME)21,9,10                                           MCOMB
   21 WRITE (6,24)                                                    MCOMB
   24 FORMAT (//10X34HTAPES ARE NOT IN THE CORRECT ORDER)             MCOMB
      CALL EXIT                                                       MCOMB
   12 END FILE KT                                                     MCOMB
      REWIND KT                                                       MCOMB
   15 REWIND I                                                        MCOMB
      XTIME = WTIME                                                   MCOMB
      WRITE (6,20) L, XTIME                                           MCOMB
```

```
      20 FORMAT(13X 4HTAPE I3, 10H ENDING AT F10.5,     29H HAS BEEN LOADED        MCOMB
        1 ON NEW TAPE./)                                                            MCOMB
      18 CONTINUE                                                                   MCOMB
         IF(KT .GT. 15) GO TO 200                                                   MCOMB
         WRITE(6,30) NTAPE, ALPHA(KT)                                               MCOMB
      30 FORMAT (1H010X9HDATA FROMI2,38H PLOT TAPES HAS BEEN COMBINED ON UN         MCOMB
        1IT A2)                                                                     MCOMB
     200 STOP                                                                       MCOMB
         END                                                                        MCOMB
```

```
MFSD      MFSD      MFSD      MFSD      MFSD      MFSD      MFSD      MFSD      MFSD      MFSD      MFSD

      SUBROUTINE MFSD(A,N,$)                                            MFSD
      DIMENSION A(1)                                                    MFSD
      DOUBLE PRECISION DPIV, DSUM                                       MFSD
C     INITIALIZE DIAGONAL-LOOP                                          MFSD
      KPIV = 0                                                          MFSD
      DO 11 K=1,N                                                       MFSD
      KPIV = KPIV + K                                                   MFSD
      IND = KPIV                                                        MFSD
      LEND = K - 1                                                      MFSD
C     START FACTORIZATION-LOOP OVER K-TH ROW                            MFSD
      DO 11 I=K,N                                                       MFSD
      DSUM = 0.D0                                                       MFSD
      IF(LEND) 2,4,2                                                    MFSD
C     START INNER LOOP                                                  MFSD
    2 DO 3 L=1,LEND                                                     MFSD
      LANF = KPIV - L                                                   MFSD
      LIND = IND - L                                                    MFSD
    3 DSUM = DSUM + A(LANF)*A(LIND)                                     MFSD
C     TRANSFORM ELEMENT A(IND)                                          MFSD
    4 DSUM = A(IND) - DSUM                                              MFSD
      IF(I-K) 10,5,10                                                   MFSD
C     TEST FOR NEGATIVE PIVOT ELEMENT AND FOR LOSS OF SIGNIFICANCE      MFSD
    5 IF(DSUM) 12,12,9                                                  MFSD
C     COMPUTE PIVOT ELEMENT                                             MFSD
    9 DPIV = DSQRT(DSUM)                                                MFSD
      A(KPIV) = DPIV                                                    MFSD
      DPIV = 1.D0/DPIV                                                  MFSD
      GO TO 11                                                          MFSD
C     CALCULATE TERMS IN ROW                                            MFSD
   10 A(IND) = DSUM*DPIV                                                MFSD
   11 IND = IND + I                                                     MFSD
      RETURN                                                            MFSD
   12 RETURN 3                                                          MFSD
      END                                                               MFSD
```

```
      FUNCTION NBLANK (WORD,N)                    00002220   NBLANK
      INTEGER  WORD(24),BLANK                                NBLANK
      DATA  BLANK/6H      /                       00002240   NBLANK
      N1 = N + 1                                  00002250   NBLANK
      DO 20 M=1,N                                 00002260   NBLANK
      I = N1 - M                                  00002270   NBLANK
      IF (WORD(I)-BLANK) 40,20,40                 00002280   NBLANK
   20 CONTINUE                                    00002290   NBLANK
   40 NBLANK = 6 * I                              00002300   NBLANK
      RETURN                                      00002310   NBLANK
      END                                         00002320   NBLANK
```

```
      SUBROUTINE NEWTMP(PR,VP,W,TMPTIM)                                 NEWTMP
C                                                                       NEWTMP
      INTEGER HEADER(12), PR(1), VP(1), W(1)                            NEWTMP
C                                                                       NEWTMP
      COMMON /FIXCON/ CON(1)                                            NEWTMP
      COMMON /TEMP  / T(1)                                              NEWTMP
      COMMON /DIMENS/ NND, NNA, NTL                                     NEWTMP
C                                                                       NEWTMP
      DATA IUT / 24 /                                                   NEWTMP
C                                                                       NEWTMP
C                                                                       NEWTMP
      READ(IUT) HEADER, (NP,I=1,6), NPR, NVP, NP,NP,NP, NW, NP,NP, NSL  NEWTMP
      IF(PR(1) .NE. NPR) GO TO 10                                       NEWTMP
      IF(VP(1) .NE. NVP) GO TO 10                                       NEWTMP
      IF(W (1) .NE. NW ) GO TO 10                                       NEWTMP
      IF(NTL   .EQ. NSL) GO TO 20                                       NEWTMP
   10 CALL TOPLIN                                                       NEWTMP
      WRITE(6,15. HEADER, PR(1), VP(1), W(1), NTL, NPR, NVP, NW, NSL    NEWTMP
   15 FORMAT(82H0* * *  ITEM COUNTS FROM HISTORY TAPE DO NOT MATCH ITEM NEWTMP
     1COUNTS FOR THIS RUN  * * *// 8X 29HTHE HISTORY TAPE LABEL IS -    NEWTMP
     2 12A6 // 8X 43HTHE ITEM COUNTS FOR THIS RUN ARE  - - - - -  I5,   NEWTMP
     3 3HNPR, I5, 3HNVP, I5, 3HNW , I5, 3HNSL /                         NEWTMP
     4          8X 43HTHE ITEM COUNTS FROM THE HISTORY TAPE ARE -  I5,  NEWTMP
     5 3HNPR, I5, 3HNVP, I5, 3HNW , I5, 3HNSL /)                        NEWTMP
      CALL WLKBCK                                                       NEWTMP
      CALL EXIT                                                         NEWTMP
C                                                                       NEWTMP
   20 READ(IUT) XTIME, (PR(I+1),I=1,NPR), (VP(I+1),I=1,NVP),            NEWTMP
     1                    (W (I+1),I=1,NW ), (T (I   ),I=1,NSL)         NEWTMP
      IF(XTIME .LT. 0.0    ) GO TO 30                                   NEWTMP
      IF(XTIME .LT. TMPTIM) GO TO 20                                    NEWTMP
      GO TO 50                                                          NEWTMP
   30 XTIME = -XTIME                                                    NEWTMP
      WRITE(6,40)                                                       NEWTMP
   40 FORMAT(80H0HISTORY TAPE READ TIME IS GREATER THAN THE LAST TIME PONEWTMP
     1INT ON THE HISTORY TAPE )                                         NEWTMP
   50 WRITE(6,60) XTIME                                                 NEWTMP
   60 FORMAT(62H0INITIAL TEMPERATURES AND VALVE POSITIONS INPUT FROM U-TNEWTMP
     1APE AT  G12.5 )                                                   NEWTMP
      RETURN                                                            NEWTMP
      END                                                               NEWTMP
```

NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN NONLIN

```
      SUBROUTINE NONLIN                                         NONLIN
      INCLUDE COMM,LIST                                         NONLIN
      NNC = NNA+NND                                             NONLIN
      DO 50 I = 1,NNC                                           NONLIN
   50 Q(I) = 0.0                                                NONLIN
      KEEP1 = NDIM                                     *NEW     NONLIN
      KEEP2 = NTH                                      *NEW     NONLIN
      NDIM = NDIM - NNC                                *NEW     NONLIN
      NTH = NTH + NNC                                  *NEW     NONLIN
      CALL VARBL1                                      *NEW     NONLIN
      NDIM = KEEP1                                     *NEW     NONLIN
      NTH = KEEP2                                      *NEW     NONLIN
      J1 = 0                                           **-1     NONLIN
      J2 = 1                                                    NONLIN
      IF(NND.EQ.0) GO TO 86                                     NONLIN
      DO 85 I = 1,NND                                           NONLIN
      INCLUDE VARC,LIST                                         NONLIN
      INCLUDE VARQ,LIST                                         NONLIN
   70 J1 = J1+1                                                 NONLIN
      LG = FLD(5,16,NSQ1(J1))                                   NONLIN
      IF(LG.EQ.0) GO TO 85                                      NONLIN
      LTA = FLD(22,14,NSQ1(J1))                                 NONLIN
      INCLUDE VARG,LIST                                         NONLIN
C     CHECK FOR LAST CONDUCTOR                                  NONLIN
      IF(NSQ1(J1).GT.0) GO TO 70                                NONLIN
   85 CONTINUE                                                  NONLIN
   86 CONTINUE                                                  NONLIN
      IF(NNA.EQ.0) GO TO 166                                    NONLIN
      L1 = NND+1                                                NONLIN
      JJ1 = J1                                                  NONLIN
      JJ2 = J2                                                  NONLIN
      DO 165 L = L1,NNC                                         NONLIN
      INCLUDE VRQ2,LIST                                         NONLIN
  135 JJ1 = JJ1+1                                               NONLIN
      LG = FLD(5,16,NSQ1(JJ1))                                  NONLIN
      LTA = FLD(22,14,NSQ1(JJ1))                                NONLIN
      INCLUDE VRG2,LIST                                         NONLIN
C     CHECK FOR LAST CONDUCTOR                                  NONLIN
      IF(NSQ1(JJ1).GT.0) GO TO 135                              NONLIN
  165 CONTINUE                                                  NONLIN
  166 CONTINUE                                                  NONLIN
      RETURN                                                    NONLIN
      END                                                       NONLIN
```

```
      SUBROUTINE NTWRK(L14)
C
      LOGICAL LVP, LIFR, LAR, LOP, COP, FIRST
C
      DIMENSION RDATA(1)
C
      COMMON /ARRAY / NDATA(1)
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9
      COMMON /FDATA / LVP, LIFR, LAR, LOP
      COMMON /FDATA / COP, LRO, NRO, RO, LMU, NMU, XMU, GC2
      COMMON /FDATA / TOL, MXPASS, EPS, FROF
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)
C
      EQUIVALENCE (RDATA,NDATA)
C
C
      L20=NDATA(L14)-3
      L25 = NTH + 1
      NEXT(L25) = NDIM
      NPRN = 0
      FIRST = .TRUE.
      EFROF = 1.0                                              *NEW
C
C PASS LOOP
C
      DO 540 NPASS=1,MXPASS
      DWRK = 0.0
C
      IF(.NOT. COP) GO TO 470
      IF(.NOT. FIRST) CALL TOPLIN
      CALL LINECK(4)
      WRITE(6,460) NPASS, NDATA(L14+1)
  460 FORMAT(///12H * * *  PASS I5, 13H FOR NETWORK A6, 7H  * * *)
C
C TUBE LOOP
C
  470 DO 520 J=4,L20,5
      K    = L14 + J
      NTB  = NDATA(K)
      NFRM = NDATA(K+1)
      NTO  = NDATA(K+2)
      KDAT = NDATA(K+3)
      L30  = NDATA(K+4)
C
      IF(FIRST) GO TO 475
      NFRM = NEXT(L25+NFRM)
      NTO = NEXT(L25+NTO)
C
  475 IF(.NOT. COP) GO TO 500
      CALL LINECK(3)
      WRITE(6,480) NTB, NFRM, NTO, KDAT, RDATA(L2+NTB)
  480 FORMAT(//              7X 7HNTB   = I10  , 8X 7HNFRM  = I10  ,
     1  8X 7HNTO   = I10  , 8X 7HKDAT  = I10  , 8X 7HR(NTB)= G13.5 )
C
  500 IF(KDAT) 505,517,510
  505 NTH = NTH + NPRN + 1
      CALL NTWRK1(L30,RDATA(L2+NTB),NFRM,NTO)
      NTH = L25 - 1
      RDATA(L4+NTB) = RDATA(L2+NTB)/(RDATA(L3+NFRM)-RDATA(L3+NTO))
```

```
      IF( .NOT. COP) GO TO 515                                          *NEW   NTWRK
      CALL LINECK(3)                                                    *NEW   NTWRK
      WRITE(6,506) NPASS,NDATA(L14+1)                                   *NEW   NTWRK
  506 FORMAT(// 23H * * *  CONTINUING PASS I5, 13H FOR NETWORK A6,      *NEW   NTWRK
     1 7H  * * *)                                                       *NEW   NTWRK
      GO TO 515                                                                NTWRK
  510 CALL FLRES(L30,NTB)                                                      NTWRK
C                                                                              NTWRK
C APPLY USER ADDED RESISTANCE TO FLOW CONDUCTOR                                NTWRK
C                                                                              NTWRK
  517 IF(LAR) RDATA(L4+NTB) = 1.0/(1.0/RDATA(L4+NTB)+RDATA(L8+NTB))            NTWRK
  515 IF(.NOT. FIRST) GO TO 520                                                NTWRK
      CALL PRN(NEXT(L25),NPRN,NDATA(K+1))                                      NTWRK
      CALL PRN(NEXT(L25),NPRN,NDATA(K+2))                                      NTWRK
  520 CONTINUE                                                                 NTWRK
C                                                                              NTWRK
      CALL FLOBAL(NPRN,L14,  0,  0,  0, EFROF,DWMX)                     *NEW   NTWRK
C                                                                       **-1   NTWRK
      IF(DWMX .GT. TOL) GO TO 530                                              NTWRK
      DO 525 J=4,L20,5                                                         NTWRK
      K = L14 + J                                                              NTWRK
      NFRM = NDATA(K+1)                                                        NTWRK
      NTO = NDATA(K+2)                                                         NTWRK
      NDATA(K+1) = NEXT(L25+NFRM)                                              NTWRK
      NDATA(K+2) = NEXT(L25+NTO)                                               NTWRK
      IF(.NOT. LOP) GO TO 525                                                  NTWRK
C                                                                              NTWRK
C CALCULATE PRESSURE DROP IN TUBE                                              NTWRK
C                                                                              NTWRK
      NTB  = NDATA(K)                                                          NTWRK
      NFRM = NDATA(K+1)                                                        NTWRK
      NTO  = NDATA(K+2)                                                        NTWRK
      RDATA(L9+NTB) = RDATA(L3+NFRM) - RDATA(L3+NTO)                           NTWRK
  525 CONTINUE                                                                 NTWRK
      RETURN                                                                   NTWRK
  530 FIRST = .FALSE.                                                          NTWRK
      EFROF = FROF                                                      *NEW   NTWRK
  540 CONTINUE                                                                 NTWRK
C                                                                              NTWRK
      CALL TOPLIN                                                              NTWRK
      WRITE(6,560) NDATA(L14+1), MXPASS, DWMX, TOL                             NTWRK
  560 FORMAT(85H0* * *  SUBROUTINE NTWRK  FAILED TO CONVERGE TO A SOLUTI       NTWRK
     1ON FOR PRESSURES FOR NETWORK  A6, 7H  * * * //                           NTWRK
     2 8X 19HMAXIMUM PASSES     - I10   /                                      NTWRK
     3 8X 19HMAXIMUM CHANGE     - G13.8 /                                      NTWRK
     4 8X 19HMAXIMUM ALLOWABLE - G13.8   )                                     NTWRK
C                                                                              NTWRK
      CALL WLKBCK                                                              NTWRK
      CALL OUTCAL                                                              NTWRK
      CALL FLPRNT(RDATA(L9),15HFLOW CONDUCTORS)                                NTWRK
      CALL EXIT                                                                NTWRK
C                                                                              NTWRK
      END                                                                      NTWRK
```

```
      SUBROUTINE NTWRK1(L14,WIN,NP1,NP3)                               NTWRK1
C                                                                       NTWRK1
      LOGICAL LVP, LIFR, LAR, LOP, COP, FIRST                           NTWRK1
C                                                                       NTWRK1
      DIMENSION RDATA(1)                                                NTWRK1
C                                                                       NTWRK1
      COMMON /ARRAY / NDATA(1)                                          NTWRK1
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9                    NTWRK1
      COMMON /FDATA / LVP, LIFR, LAR, LOP                               NTWRK1
      COMMON /FDATA / COP, LRJ, NRJ, RJ, LMU, NMU, XMU, GC2             NTWRK1
      COMMON /FDATA / TOL, MXPASS, EPS, FRDF                            NTWRK1
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)                                NTWRK1
C                                                                       NTWRK1
      EQUIVALENCE (RDATA,NDATA)                                         NTWRK1
C                                                                       NTWRK1
C                                                                       NTWRK1
      L20=NDATA(L14)-3                                                  NTWRK1
      L25 = NTH + 1                                                     NTWRK1
      NEXT(L25) = NDIM                                                  NTWRK1
      NPRN = 0                                                          NTWRK1
      FIRST = .TRUE.                                                    NTWRK1
      EFRDF = 1.0                                                  *NEW NTWRK1
C                                                                       NTWRK1
C PASS LOOP                                                             NTWRK1
C                                                                       NTWRK1
      DO 540 NPASS=1,MXPASS                                             NTWRK1
      DWMX = 0.0                                                        NTWRK1
C                                                                       NTWRK1
      IF(.NOT. COP) GO TO 470                                           NTWRK1
      IF(.NOT. FIRST) CALL TOPLIN                                       NTWRK1
      CALL LINECK(4)                                                    NTWRK1
      WRITE(6,460) NPASS, NDATA(L14+1)                                  NTWRK1
  460 FORMAT(///12H * * *  PASS I5, 13H FOR NETWORK A6, 7H  * * *)      NTWRK1
C                                                                       NTWRK1
C TUBE LOOP                                                             NTWRK1
C                                                                       NTWRK1
  470 DO 520 J=4,L20,5                                                  NTWRK1
      K    = L14 + J                                                    NTWRK1
      NTB  = NDATA(K)                                                   NTWRK1
      NFRM = NDATA(K+1)                                                 NTWRK1
      NTO  = NDATA(K+2)                                                 NTWRK1
      KDAT = NDATA(K+3)                                                 NTWRK1
      L30  = NDATA(K+4)                                                 NTWRK1
C                                                                       NTWRK1
      IF(FIRST) GO TO 475                                               NTWRK1
      NFRM = NEXT(L25+NFRM)                                             NTWRK1
      NTO = NEXT(L25+NTO)                                               NTWRK1
C                                                                       NTWRK1
  475 IF(.NOT. COP) GO TO 500                                           NTWRK1
      CALL LINECK(3)                                                    NTWRK1
      WRITE(6,480) NTB, NFRM, NTO, KDAT, RDATA(L2+NTB)                  NTWRK1
  480 FORMAT(//            7X 7HNTB   = I10  , 8X 7HNFRM  = I10  ,      NTWRK1
     1  8X 7HNTO   = I10  , 8X 7HKDAT  = I10  , 8X 7HL(NTB)= G13.8 )    NTWRK1
C                                                                       NTWRK1
  500 IF(KDAT) 505,517,510                                              NTWRK1
  505 NTH = NTH + NPRN + 1                                              NTWRK1
      CALL NTWRKN(L30,RDATA(L2+NTB),NFRM,NTO)                           NTWRK1
      NTH = L25 - 1                                                     NTWRK1
      RDATA(L4+NTB) = RDATA(L2+NTB)/(RDATA(L3+NFRM)-RDATA(L3+NTO))      NTWRK1
```

A-79

```
      IF( .NOT. COP) GO TO 515                                          *NEW    NTWRK1
      CALL LINECK(3)                                                    *NEW    NTWRK1
      WRITE(6,506) NPASS,NDATA(L14+1)                                   *NEW    NTWRK1
  506 FORMAT(// 23H * * *  CONTINUING PASS I5, 13H FOR NETWORK A6,      *NEW    NTWRK1
     1 7H  * * *)                                                       *NEW    NTWRK1
      GO TO 515                                                                 NTWRK1
  510 CALL FLRES(L30,NTB)                                                       NTWRK1
C                                                                               NTWRK1
C APPLY USER ADDED RESISTANCE TO FLOW CONDUCTOR                                 NTWRK1
C                                                                               NTWRK1
  517 IF(LAR) RDATA(L4+NTB) = 1.0/(1.0/RDATA(L4+NTB)+RDATA(L8+NTB))             NTWRK1
  515 IF(.NOT. FIRST) GO TO 520                                                 NTWRK1
      CALL PRN(NEXT(L25),NPRN,NDATA(K+1))                                       NTWRK1
      CALL PRN(NEXT(L25),NPRN,NDATA(K+2))                                       NTWRK1
  520 CONTINUE                                                                  NTWRK1
C                                                                               NTWRK1
      CALL FLOBAL(NPRN,L14,WIN,NPI,NPO, EFROF,DWMX)                     *NEW    NTWRK1
C                                                                       **-1    NTWRK1
      IF(DWMX .GT. TOL) GO TO 530                                               NTWRK1
      DO 525 J=4,L20,5                                                          NTWRK1
      K = L14 + J                                                               NTWRK1
      NFRM = NDATA(K+1)                                                         NTWRK1
      NTO = NDATA(K+2)                                                          NTWRK1
      NDATA(K+1) = NEXT(L25+NFRM)                                               NTWRK1
      NDATA(K+2) = NEXT(L25+NTO)                                                NTWRK1
      IF(.NOT. LOP) GO TO 525                                                   NTWRK1
C                                                                               NTWRK1
C CALCULATE PRESSURE DROP IN TUBE                                               NTWRK1
C                                                                               NTWRK1
      NTB  = NDATA(K)                                                           NTWRK1
      NFRM = NDATA(K+1)                                                         NTWRK1
      NTO  = NDATA(K+2)                                                         NTWRK1
      RDATA(L9+NTB) = RDATA(L3+NFRM) - RDATA(L3+NTO)                            NTWRK1
  525 CONTINUE                                                                  NTWRK1
      RETURN                                                                    NTWRK1
  530 FIRST = .FALSE.                                                           NTWRK1
      EFROF = FROF                                                      *NEW    NTWRK1
  540 CONTINUE                                                                  NTWRK1
C                                                                               NTWRK1
      CALL TOPLIN                                                               NTWRK1
      WRITE(6,560) NDATA(L14+1), MXPASS, DWMX, TOL                              NTWRK1
  560 FORMAT(85H0* * *  SUBROUTINE NTWRK1 FAILED TO CONVERGE TO A SOLUTI        NTWRK1
     1ON FOR PRESSURES FOR NETWORK  A6, 7H  * * * //                            NTWRK1
     2 8X 19HMAXIMUM PASSES    - I10   /                                        NTWRK1
     3 8X 19HMAXIMUM CHANGE    - G13.8 /                                        NTWRK1
     4 8X 19HMAXIMUM ALLOWABLE - G13.8   )                                      NTWRK1
C                                                                               NTWRK1
      CALL WLKBCK                                                               NTWRK1
      CALL OUTCAL                                                               NTWRK1
      CALL FLPRNT(RDATA(L4),15HFLOW CONDUCTORS)                                 NTWRK1
      CALL EXIT                                                                 NTWRK1
C                                                                               NTWRK1
      END                                                                       NTWRK1
```

```
      SUBROUTINE NTWRKN(L14,MIN,NP1,NP2)
C
      LOGICAL LVP, LIFR, LAR, LDP, COP, FIRST
C
      DIMENSION RDATA(1)
C
      COMMON /ARRAY / NDATA(1)
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9
      COMMON /FDATA / LVP, LIFR, LAR, LDP
      COMMON /FDATA / COP, LRO, NRO, RO, LMU, NMU, XMU, GC2
      COMMON /FDATA / TOL, MXPASS, EPS, FROF
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)
C
      EQUIVALENCE (RDATA,NDATA)
C
C
      L20=NDATA(L14)-3
      L25 = NTH + 1
      NEXT(L25) = NDIM
      NPRN = 0
      FIRST = .TRUE.
      EFROF = 1.0                                              *NEW
C
C PASS LOOP
C
      DO 540 NPASS=1,MXPASS
      DWMX = 0.0
C
      IF(.NOT. COP) GO TO 470
      IF(.NOT. FIRST) CALL TOPLIN
      CALL LINECK(4)
      WRITE(6,460) NPASS, NDATA(L14+1)
  460 FORMAT(///12H * * *  PASS I5, 13H FOR NETWORK A6, 7H  * * *)
C
C TUBE LOOP
C
  470 DO 520 J=4,L20,5
      K    = L14 + J
      NTB  = NDATA(K)
      NFRM = NDATA(K+1)
      NTO  = NDATA(K+2)
      KDAT = NDATA(K+3)
      L30  = NDATA(K+4)
C
      IF(FIRST) GO TO 475
      NFRM = NEXT(L25+NFRM)
      NTO = NEXT(L25+NTO)
C
  475 IF(.NOT. COP) GO TO 500
      CALL LINECK(3)
      WRITE(6,480) NTB, NFRM, NTO, KDAT, RDATA(L2+NTB)
  480 FORMAT(//               7X 7HNTB    = I10  , 8X 7HNFRM  = I10  ,
     1  8X 7HNTO    = I10  , 8X 7HKDAT  = I10  , 8X 7HW(NTB)= G13.8 )
C
  500 IF(KDAT) 505,517,510
C
  505 CALL TOPLIN
      WRITE(6,506) NDATA(L14+1)
  506 FORMAT(16HO* * *  NETWORK A6, 37H MUST NOT CONTAIN A SUBNETWORK  *
```

```
     1 * *)                                                        NTWRKN
      CALL WLKBCK                                                  NTWRKN
      CALL EX.T                                                    NTWRKN
C                                                                  NTI
  510 CALL FLRES(L30,NTB)                                          NTL
C                                                                  NTWRKN
C APPLY USER ADDED RESISTANCE TO FLOW CONDUCTOR                     NTWRKN
C                                                                  NTWRKN
  517 IF(LAR) RDATA(L4+NTB) = 1.0/(1.0/RDATA(L4+NTB)+RDATA(L8+NTB)) NTWRKN
  515 IF(.NOT. FIRST) GO TO 520                                    NTWRKN
      CALL PRN(NEXT(L25),NPRN,NDATA(K+1))                          NTWRKN
      CALL PRN(NEXT(L25),NPRN,NDATA(K+2))                          NTWRKN
  520 CONTINUE                                                     NTWRKN
C                                                                  NTWRKN
      CALL FLOBAL(NPRN,L14,WIN,NPI,NPO, EFROF,DWMX)          *NEW  NTWRKN
C                                                            **-1  NTWRKN
      IF(DWMX .GT. TOL) GO TO 530                                  NTWRKN
      DO 525 J=4,L20,5                                             NTWRKN
      K = L14 + J                                                  NTWRKN
      NFRM = NDATA(K+1)                                            NTWRKN
      NTO = NDATA(K+2)                                             NTWRKN
      NDATA(K+1) = NEXT(L25+NFRM)                                  NTWRKN
      NDATA(K+2) = NEXT(L25+NTO)                                   NTWRKN
      IF(.NOT. LOP) GO TO 525                                      NTWRKN
C                                                                  NTWRKN
C CALCULATE PRESSURE DROP IN TUBE                                  NTWRKN
C                                                                  NTWRKN
      NTB  = NDATA(K)                                              NTWRKN
      NFRM = NDATA(K+1)                                            NTWRKN
      NTO  = NDATA(K+2)                                            NTWRKN
      RDATA(L9+NTB) = RDATA(L3+NFRM) - RDATA(L3+NTO)               NTWRKN
  525 CONTINUE                                                     NTWRKN
      RETURN                                                       NTWRKN
  530 FIRST = .FALSE.                                              NTWRKN
      EFROF = FROF                                           *NEW  NTWRKN
  540 CONTINUE                                                     NTWRKN
C                                                                  NTWRKN
      CALL TOPLIN                                                  NTWRKN
      WRITE(6,560) NDATA(L14+1), MXPASS, DWMX, TOL                 NTWRKN
  560 FORMAT(85H0* * *  SUBROUTINE NTWRKN FAILED TO CONVERGE TO A SOLUTI NTWRKN
     1ON FOR PRESSURES FOR NETWORK  A6, 7H  * * * //               NTWRKN
     2 8X 19HMAXIMUM PASSES     - I10   /                          NTWRKN
     3 8X 19HMAXIMUM CHANGE     - G13.8 /                          NTWRKN
     4 8X 19HMAXIMUM ALLOWABLE  - G13.8  )                         NTWRKN
C                                                                  NTWRKN
      CALL WLKBCK                                                  NTWRKN
      CALL OUTCAL                                                  NTWRKN
      CALL FLPRNT(RDATA(L4),15HFLOW CONDUCTORS)                    NTWRKN
      CALL EXIT                                                    NTWRKN
C                                                                  NTWRKN
      END                                                          NTWRKN
```

```
      SUBROUTINE PFCS(MLOC,NLOC,NAME)                                    PFCS
C                                                                        PFCS
      LOGICAL LVP, LIFR, LAR, LOP, COP, LRO, LMU, LPASS, LPUMP           PFCS
C                                                                        PFCS
      DIMENSION MLOC(1), NLOC(1), NAME(9)                                PFCS
      DIMENSION RDATA(1), EXT(1)                                         PFCS
C                                                                        PFCS
      COMMON /ARRAY / NDATA(1)                                           PFCS
      COMMON /TEMP  / T    (1)                                           PFCS
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)                                 PFCS
      COMMON /FIXCON/ CON(1)                                             PFCS
      COMMON /FDATA / L2, L3, L4, L5, L6, L7, L8, L9                     PFCS
      COMMON /FDATA / LVP, LIFR, LAR, LOP                                PFCS
      COMMON /FDATA / COP, LRO, NRO, RO, LMU, NMU, XMU, GC2              PFCS
      COMMON /FDATA / TOL, MXPASS, EPS, FRDF                             PFCS
      COMMON/POINTN/LNODE                                                PFCS
C                                                                        PFCS
      EQUIVALENCE (RDATA,NDATA), (EXT,NEXT)                              PFCS
      EQUIVALENCE (CON(1),TIMEN), (CON(2),DTIMEU), (CON(28),LC)          PFCS
      EQUIVALENCE (TSEN,NSEN), (TSET,NSET)                               PFCS
C                                                                        PFCS
      DATA MAXI /65000/                                                  PFCS
C                                                                        PFCS
C                                                                        PFCS
      LVP  = .FALSE.                                                     PFCS
      LIFR = .FALSE.                                                     PFCS
      LAR  = .FALSE.                                                     PFCS
      LOP  = .FALSE.                                                     PFCS
      IF(MLOC(1) .EQ. 8) GO TO 20                                        PFCS
      CALL TOPLIN                                                        PFCS
      WRITE(6,10) MLOC(1), NAME                                          PFCS
   10 FORMAT(71H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR  PFCS
     1FLOW DATA, IC = I5, 7H  * * * // 8X 4HFOR  9A6)                    PFCS
      GO TO 140                                                          PFCS
C                                                                        PFCS
   20 L2 = MLOC(2)                                                       PFCS
      IF(NDATA(L2) .GT. 0) GO TO 40                                      PFCS
      CALL TOPLIN                                                        PFCS
      WRITE(6,30) NDATA(L2), NAME                                        PFCS
   30 FORMAT(72H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR  PFCS
     1FLOW RATES, IC = I5, 7H  * * * // 8X 4HFOR  9A6)                   PFCS
      GO TO 140                                                          PFCS
C                                                                        PFCS
   40 L3 = MLOC(3)                                                       PFCS
      IF(NDATA(L3) .GT. 0) GO TO 60                                      PFCS
      CALL TOPLIN                                                        PFCS
      WRITE(6,50) NDATA(L3), NAME                                        PFCS
   50 FORMAT(72H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR  PFCS
     1PRESSURES, IC =  I5, 7H  * * * // 8X 4HFOR  9A6)                   PFCS
      GO TO 140                                                          PFCS
C                                                                        PFCS
   60 L4 = MLOC(4)                                                       PFCS
      IF(NDATA(L4) .EQ. NDATA(L2)) GO TO 80                              PFCS
      CALL TOPLIN                                                        PFCS
      WRITE(6,70) NDATA(L4), NAME                                        PFCS
   70 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR  PFCS
     1FLOW CONDUCTORS, IC = I5, 7H  * * * // 8X 4HFOR  9A6)              PFCS
      GO TO 140                                                          PFCS
C                                                                        PFCS
```

```
   80 L5 = MLOC(5)
      IF(L5 .LT. 1) GO TO 100
      IF(NDATA(L5) .GT. 0) LVP = .TRUE.
  100 L6 = MLOC(6)
      IF(L6 .LT. 1) GO TO 120
      IF(NDATA(L6) .EQ. NDATA(L3)) LIFR = .TRUE.
  120 L7 = MLOC(7)
      IF(MOD(NDATA(L7),6) .EQ. 0) GO TO 160
      CALL TOPLIN
      WRITE(6,130) NDATA(L7), NAME
  130 FORMAT(77H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR
     1FLUID TYPE DATA, IC = I5, 7H  * * * // 8X 4HFOR  9A6)
  140 WRITE(6,150) (MLOC(I),I=2,9)
  150 FORMAT(5H0ARW = I6, 6H APR = I6, 6H ACF = I6, 6H AVP = I6,
     1 7H AIFR = I6, 6H AFT = I6, 6H AAR = I6, 6H ADP = I6)
      CALL WLKBCK
      CALL EXIT
C
  160 L8 = MLOC(8)
      IF(L8 .LT. 1) GO TO 162
      IF(NDATA(L8) .EQ. NDATA(L2)) LAR = .TRUE.
  162 L9 = MLOC(9)
      IF(L9 .LT. 1) GO TO 166
      IF(NDATA(L9) .EQ. NDATA(L2)) LDP = .TRUE.
  166 IF(NLOC(1) .EQ. 6) GO TO 180
      CALL TOPLIN
      WRITE(6,170) NLOC(1), NAME
  170 FORMAT(78H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR
     1FLOW SYSTEM DATA, IC = I5, 7H  * * * // 8X 4HFOR  9A6)
      CALL WLKBCK
      CALL EXIT
C
  180 L12 = NLOC(2)
      IF(NDATA(L12) .EQ. 5) GO TO 200
      CALL TOPLIN
      WRITE(6,190) NDATA(L12), NAME
  190 FORMAT(79H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR
     1SYSTEM PARAMETERS, IC = I5, 7H  * * * // 8X 4HFOR  9A6)
      GO TO 220
  200 L13 = NLOC(3)
      IF(NDATA(L13) .EQ. 4) GO TO 240
      CALL TOPLIN
      WRITE(6,210) NDATA(L13), NAME
  210 FORMAT(81H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR
     1SOLUTION PARAMETERS, IC = I5, 7H  * * * // 8X 4HFOR  9A6)
  220 WRITE(6,230) (NLOC(I),I=2,6)
  230 FORMAT(7H0ASYS = I6, 7H ASOL = I6, 7H ANET = I6, 6H AVL = I6,
     1 5H AP = I6, 6H KOP = I5)
      CALL WLKBCK
      CALL EXIT
C
  240 L14 = NLOC(4 )
  260 L15 = NLOC(5 )
  280 L16 = NLOC(6 )
C
C SYSTEM PARAMETERS
C
  300 COP = .FALSE.
      IF(NLOC(7) .NE. 0) COP = .TRUE.
      IF(COP .AND. LNODE .EQ. 0) CALL NNREAD(1)
      NRO = NDATA(L12+2)
      IF(NRO .LT. 1 .OR. NRO .GT. MAXI) GO TO 310
      LRO = .TRUE.
```

```
      GO TO 320                                                         PFCS
  310 LRO = .FALSE.                                                     PFCS
      RO  = RDATA(L12+2)                                                PFCS
  320 NMU = NDATA(L12+3)                                                PFCS
      IF(NMU .LT. 1 .OR. NMU .GT. MAXI) GO TO 330                       PFCS
      LMU = .TRUE.                                                      PFCS
      NMU = NDATA(L12+3)                                                PFCS
      GO TO 340                                                         PFCS
  330 LMU = .FALSE.                                                     PFCS
      XMU = RDATA(L12+3)                                                PFCS
  340 GC2 = RDATA(L12+5)*2.0                                            PFCS
C                                                                       PFCS
C SOLUTION PARAMETERS                                                   PFCS
C                                                                       PFCS
      TOL    = RDATA(L13+1)                                             PFCS
      MXPASS = NDATA(L13+2)                                             PFCS
      EPS    = RDATA(L13+3)                                             PFCS
      FROF   = RDATA(L13+4)                                             PFCS
C                                                                       PFCS
C  VALVES                                                               PFCS
C                                                                       PFCS
      IF(L15 .LT. 1) GO TO 510                                          PFCS
      IF(NDATA(L15) .LT. 1) GO TO 510                                   PFCS
      IF(LVP) GO TO 348                                                 PFCS
  343 WRITE(6,344) NDATA(L5), NAME                                      PFCS
  344 FORMAT(77H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR   PFCS
     1VALVE POSITIONS, IC = I5, 7H  * * * // 8X 4HFOR  9A6)             PFCS
      CALL WLKBCK                                                       PFCS
      CALL EXIT                                                         PFCS
C                                                                       PFCS
  348 L40 = NDATA(L15)                                                  PFCS
      DO 500 J=1,L40                                                    PFCS
      L41   = NDATA(L15+J)                                              PFCS
      NV    = NDATA(L41+1)                                              PFCS
      MODE  = NDATA(L41+4)                                              PFCS
      IF(MODE .EQ. 0) GO TO 500                                         PFCS
      IF(NV .GT. NDATA(L5)) GO TO 343                                   PFCS
      XMINI = RDATA(L41+5)                                              PFCS
      XMAXI = RDATA(L41+6)                                              PFCS
      NSEN  = NDATA(L41+8)                                              PFCS
      IC    = NDATA(L41)                                                PFCS
      IF(IC .EQ. 10) GO TO 450                                          PFCS
      NSET  = NDATA(L41+9)                                              PFCS
      IF(NSEN .GT. 0 .AND. NSEN .LT. 10000) TSEN = T(NSEN)              PFCS
      IF(NSET .GT. 0 .AND. NSET .LT. 10000) TSET = T(NSET)              PFCS
      IF(IC .EQ. 12) GO TO 360                                          PFCS
      IF(IC .EQ. 16) GO TO 420                                          PFCS
C                                                                       PFCS
      CALL TOPLIN                                                       PFCS
      WRITE(6,350) IC, NAME                                             PFCS
  350 FORMAT(72H0* *  INCORRECT NUMBER OF ELEMENTS INPUT TO PFCS  FOR   PFCS
     1VALVE DATA, IC = I5, 7H  * * * // 8X 4HFOR  9A6)                  PFCS
      CALL WLKBCK                                                       PFCS
      CALL EXIT                                                         PFCS
C                                                                       PFCS
C  RATE LIMITED                                                         PFCS
C                                                                       PFCS
  360 TDB = RDATA(L41+10)                                               PFCS
      IF(ABS(TSEN-TSET) - TDB) 500,500,380                              PFCS
  380 IF(TSEN .GT. TSET + TDB) GO TO 400                                PFCS
      XDOT = AMAX1(RDATA(L41+11)*(TSEN-TSET-TDB),-RDATA(L41+12))        PFCS
      RDATA(L5+NV) = AMAX1(RDATA(L5+NV)+XDOT*DTIMEU,XMINI)              PFCS
      GO TO 500                                                         PFCS
```

```
  400 XDOT = AMIN1(RDATA(L41+11)*(TSEN-TSET+TDB), RDATA(L41+12))        PFCS
      RDATA(L5+NV) = AMIN1(RDATA(L5+NV)+XDOT*DTIMEU,XMAXI)              PFCS
      GO TO 500                                                         PFCS
C                                                                       PFCS
C  POLYNOMIAL                                                           PFCS
C                                                                       PFCS
  420 DT = TSEN - TSET                                                  PFCS
      XSS = RDATA(L41+10) + DT*(RDATA(L41+11) + DT*(RDATA(L41+12) +     PFCS
     1                       DT*(RDATA(L41+13) + DT*(RDATA(L41+14) +    PFCS
     2                       DT*(RDATA(L41+15))))))                     PFCS
      IF(XSS - XMINI) 425,440,430                                       PFCS
  425 XSS = RDATA(L41+5)                                                PFCS
      GO TO 440                                                         PFCS
  430 IF(XSS - XMAXI) 440,440,435                                       PFCS
  435 XSS = RDATA(L41+6)                                                PFCS
  440 RDATA(L5+NV) = XSS + (RDATA(L5+NV)-XSS)*EXP(-DTIMEU/RDATA(L41+16))PFCS
      GO TO 500                                                         PFCS
C                                                                       PFCS
C SWITCHING                                                             PFCS
C                                                                       PFCS
  450 IF(IFIX((2.0*T(NSEN)-RDATA(L41+9)-RDATA(L41+10))/(RDATA(L41+9)    PFCS
     1 - RDATA(L41+10)))) 460,500,480                                   PFCS
  460 RDATA(L5+NV) = XMINI                                              PFCS
      GO TO 500                                                         PFCS
  480 RDATA(L5+NV) = XMAXI                                              PFCS
C                                                                       PFCS
  500 CONTINUE                                                          PFCS
C                                                                       PFCS
C CHECK PUMP OPTION                                                     PFCS
C                                                                       PFCS
  510 IF(L16 .LT. 1) GO TO 540                                          PFCS
      IF(NDATA(L16) - 2) 540,520,560                                    PFCS
  520 NPI = NDATA(L16+1)                                                PFCS
      NPUMP = NDATA(L16+2)                                              PFCS
      CALL DIDEGI(TIMEN,NDATA(NPUMP),RDATA(L6+NPI))                     PFCS
  540 LPUMP = .FALSE.                                                   PFCS
      GO TO 600                                                         PFCS
  560 LPUMP = .TRUE.                                                    PFCS
      NPI = NDATA(L16+1)                                                PFCS
      NPO = NDATA(L16+2)                                                PFCS
  575 IF(NDATA(L16) .GT. 3) GO TO 580                          *-12     PFCS
      KPUMP = 1                                                         PFCS
      NPUMP = NDATA(L16+3)                                              PFCS
      WMX = RDATA(NPUMP+NP-1)                                           PFCS
      DPMX = RDATA(NPUMP+2)                                             PFCS
      GO TO 590                                                         PFCS
  580 KPUMP = 2                                                         PFCS
      A0 = RDATA(L16+3)                                                 PFCS
      A1 = RDATA(L16+4)                                                 PFCS
      A2 = RDATA(L16+5)                                                 PFCS
      A3 = RDATA(L16+6)                                                 PFCS
      A4 = RDATA(L16+7)                                                 PFCS
C                                                                       PFCS
C SYSTEM SOLUTION                                                       PFCS
C                                                                       PFCS
  590 LPASS = .FALSE.                                                   PFCS
  600 DO 960 KPASS=1,20                                                 PFCS
C                                                                       PFCS
      IF(.NOT. COP) GO TO 640                                           PFCS
      CALL TOPLIN                                                       PFCS
      WRITE(6,620) KPASS, NAME                                          PFCS
  620 FORMAT(70HO* * *  CHECKOUT PRINT FOR PRESSURE/FLOW COMPUTATION SUB PFCS
     1ROUTINE  * * * // 8X 7HKPASS = I3, 5H FOR  9A6)                   PFCS
```

```
      LC = LC + 4                                                  PFCS
      IF(LPASS) GO TO 640                                          PFCS
      WRITE(6,630) TOL, MXPASS, EPS, FROF                          PFCS
  630 FORMAT(1H0 10X 5HTOL = G10.5, 9H MXPASS = I5, 6H EPS = G10.5,  PFCS
     1  7H FROF = G10.5)                                           PFCS
      LC = LC + 2                                                  PFCS
C                                                                  PFCS
  640 CALL NTWRK(L14)                                              PFCS
C                                                                  PFCS
      IF(.NOT. LPUMP) GO TO 1000                                   PFCS
C                                                                  PFCS
C PUMP                                                             PFCS
C                                                                  PFCS
      WS = RDATA(L6+NPI)                                   *NEW    PFCS
      TEST = 0.001*WS                                      **-1    PFCS
      DPS = RDATA(L3+NPI) - RDATA(L3+NPO)                          PFCS
      WK  = WS                                                     PFCS
      DPK = DPS                                                    PFCS
      GO TO (660,800), KPUMP                                       PFCS
C                                                                  PFCS
C TABULATED PUMP CURVE                                             PFCS
C                                                                  PFCS
  660 IF(LPASS) GO TO 665                                          PFCS
      C = DPS/WS                                                   PFCS
      D = 0.0                                                      PFCS
      GO TO 670                                                    PFCS
  665 C = (DPS-DPL)/(WS-WL)                                        PFCS
      D = DPL - WL*C                                               PFCS
  670 NP = NDATA(NPUMP)                                            PFCS
      IF(.NOT. COP) GO TO 690                              *NEW    PFCS
      CALL LINECK(2)                                       *NEW    PFCS
      WRITE(6,680)                                         *NEW    PFCS
  680 FORMAT(1H0 7X 39HCHECKOUT PRINT FOR TABULATED PUMP CURVE )   *NEW    PFCS
  690 DO 740 J=1,100                                       *NEW    PFCS
      WA = AMIN1(WS,WMX)                                   **-1    PFCS
      DPB = AMIN1(DPS,DPMX)                                        PFCS
      CALL DIDEG1(WA,NDATA(NPUMP),DPA)                             PFCS
      CALL REVPOL(DPB,NDATA(NPUMP),WB)                             PFCS
      IF(.NOT. COP) GO TO 710                              *NEW    PFCS
      CALL LINECK(3)                                       *NEW    PFCS
      WRITE(6,705) J, WS, WA, WB, DPS, DPA, DPB            *NEW    PFCS
  705 FORMAT(/               7X 7HJ      = I10  , 8X 7HWS    = G13.8,  *NEW    PFCS
     1  5X 7HWA    = G13.8, 5X 7HWB    = G13.8/32X 7HDPS   = G13.8,  *NEW    PFCS
     2  5X 7HDPA   = G13.8, 5X 7HDPB   = G13.8)            *NEW    PFCS
  710 A= (DPB-DPA)/(WB-WA)                                 *NEW    PFCS
      B = DPA - WA*A                                       **-1    PFCS
      WNEW = (D-B)/(A-C)                                           PFCS
      IF(ABS(WNEW-WS) - TEST) 940,940,720                          PFCS
  720 WS = WNEW                                                    PFCS
      DPS = A*WS + B                                               PFCS
  740 CONTINUE                                                     PFCS
      CALL TOPLIN                                                  PFCS
      WRITE(6,760) NAME                                            PFCS
  760 FORMAT(77H0* * *  SUBROUTINE PFCS FAILED TO CONVERGE TO A SOLUTION  PFCS
     1 FOR FLOW RATE  * * * // 8X 4HFOR  9A6)                      PFCS
      WRITE(6,770)                                                 PFCS
  770 FORMAT(//8X 52HSYSTEM TOTAL FLOW RATE IS SUPPLIED BY AN INPUT CURV  PFCS
     1E)                                                           PFCS
      GO TO 900                                                    PFCS
C                                                                  PFCS
C POLYNOMIAL PUMP CURVE                                            PFCS
C                                                                  PFCS
  800 CHECK = 0.001*DPS                                            PFCS
```

```
      A00 = A0                                                          PFCS
      A11 = A1 - DPS/WS                                                 PFCS
      WNEW = WS                                                         PFCS
      IF(.NOT. LPASS) GO TO 820                                         PFCS
      TEMP = (DPS-DPL)/(WS-WL)                                          PFCS
      A00  = A0 - DPL + TEMP*WL                                         PFCS
      A11  = A1 - TEMP                                                  PFCS
  820 DO 860 J=1,100                                                    PFCS
      FWNEW = A00 + WNEW*(A11 + WNEW*(A2 + WNEW*(A3 + WNEW*A4)))        PFCS
      IF(ABS(FWNEW) - CHECK) 940,940,840                                PFCS
  840 FP = A11 + WNEW*(2.0*A2 + WNEW*(3.0*A3 + WNEW*4.0*A4))            PFCS
      WNEW = WNEW - FWNEW/FP                                            PFCS
  860 CONTINUE                                                          PFCS
C                                                                       PFCS
      CALL TOPLIN                                                       PFCS
      WRITE(6,760) NAME                                                 PFCS
      WRITE(6,880)                                                      PFCS
  880 FORMAT(//8X 89HSYSTEM TOTAL PRESSURE DROP IS SUPPLIED BY A FOURTH PFCS
     1ORDER POLYNOMIAL FUNCTION OF FLOW RATE )                          PFCS
  900 CALL WLKBCK                                                       PFCS
      CALL OUTCAL                                                       PFCS
      CALL EXIT                                                         PFCS
C                                                                       PFCS
  940 IF(ABS(WK-WNEW) - TEST) 1000,1000,950                             PFCS
  950 RDATA(L6+NPI) = WNEW                                              PFCS
      LPASS = .TRUE.                                                    PFCS
      WL = WK                                                           PFCS
      DPL = DPK                                                         PFCS
  960 CONTINUE                                                          PFCS
C                                                                       PFCS
      CALL TOPLIN                                                       PFCS
      WRITE(6,980) NAME                                                 PFCS
  980 FORMAT(1194H0* * *  SUBROUTINE PFCS FAILED TO CONVERGE TO A SOLUTIOPFCS
     1N TO TRUE SYSTEM CHARACTERISTICS AND TRUE PUMP CURVE  * * * //    PFCS
     2 8X 4HFOR  9A6)                                                   PFCS
      CALL WLKBCK                                                       PFCS
      CALL OUTCAL                                                       PFCS
      CALL EXIT                                                         PFCS
C                                                                       PFCS
 1000 RETURN                                                            PFCS
      END                                                               PFCS
```

```
      SEE GEPLOT-*(COMBIN,NBLANK-GOPLOT-OBAL)                    PLOTA
   USE GOPLOT/CODE                                               PLOTA
```

POL POL POL POL POL POL POL POL POL POL POL

```
      FUNCTION POL(LOC,X)                                        POL
C                                                                POL
      COMMON /ARRAY/ NDATA(1)                                    POL
C                                                                POL
      CALL DIDEG1(X,NDATA(LOC),Y)                                POL
      POL = Y                                                    POL
      RETURN                                                     POL
      END                                                        POL
```

PRN PRN PRN PRN PRN PRN PRN PRN PRN PRN PRN

```
      SUBROUTINE PRN(LOC,N,NODE)                                        PRN
C                                                                       PRN
      DIMENSION LOC(1)                                                  PRN
C                                                                       PRN
C                                                                       PRN
      IF(N .LT. 1) GO TO 20                                             PRN
      DO 10 J=1,N                                                       PRN
      IF(LOC(J+1) .EQ. NODE) GO TO 30                                   PRN
   10 CONTINUE                                                          PRN
   20 N = N + 1                                                         PRN
      IF(N .GT. LOC(1)) GO TO 40                                        PRN
      LOC(N+1) = NODE                                                   PRN
      NODE = N                                                          PRN
      RETURN                                                            PRN
   30 NODE = J                                                          PRN
      RETURN                                                            PRN
C                                                                       PRN
   40 NEED = N - LOC(1)                                                 PRN
      CALL TOPLIN                                                       PRN
      WRITE(6,50) NEED                                                  PRN
   50 FORMAT(83H0* * *  INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR FLOW PRN
     1BALANCING SUBROUTINE  * * * // 8X 5HSHORT I5, 10H LOCATIONS)      PRN
      CALL WLKBCK                                                       PRN
      CALL EXIT                                                         PRN
C                                                                       PRN
      END                                                               PRN
```

```
PSOR    PSOR    PSOR    PSOR    PSOR    PSOR    PSOR    PSOR    PSOR    PSOR    PSOR

      SUBROUTINE PSOR(*,M,A,R,X,EPS,W)                                          PSOR
C                                                                               PSOR
C SUBROUTINE PSOR SOLVES A SYSTEM OF SIMULTANEOUS EQUATIONS USING A             PSOR
C      STATIONARY POINT ITERATIVE SUCCESSIVE OVERRELAXATION METHOD.             PSOR
C                                                                               PSOR
      DIMENSION A(M,M), R(M), X(M)                                              PSOR
C                                                                               PSOR
      MAXK=100*M                                                                PSOR
      W1   = W - 1.0                                                            PSOR
C                                                                               PSOR
      DO 600 K=1,MAXK                                                           PSOR
      BIGC = 0.0                                                                PSOR
      DO 500 I=1,M                                                              PSOR
      SUM = -A(I,I)*X(I)                                                        PSOR
      DO 100 J=1,M                                                              PSOR
      IF(A(I,J)) ,100                                                           PSOR
      SUM = SUM + A(I,J)*X(J)                                                   PSOR
  100 CONTINUE                                                                  PSOR
  400 TEMP = W*(R(I)-SUM)/A(I,I) - W1*X(I)                                      PSOR
      CHNG = ABS(TEMP-X(I))                                                     PSOR
      IF(CHNG .GT. BIGC) BIGC = CHNG                                            PSOR
      X(I) = TEMP                                                               PSOR
  500 CONTINUE                                                                  PSOR
C                                                                               PSOR
      IF(BIGC-EPS)700,700                                                       PSOR
  600 CONTINUE                                                                  PSOR
C                                                                               PSOR
C                                                                               PSOR
  690 CONTINUE                                                                  PSOR
C                                                                               PSOR
      WRITE(6,630) MAXK,BIGC,EPS,W                                              PSOR
  630 FORMAT(1H1 // 1X131(1H*) // 50H SUBROUTINE PSOR FAILED TO CONVERGE        PSOR
     X TO A SOLUTION. //11X 26HMAXIMUM ITERATIONS         - I10   /             PSOR
     X                  11X 26HLARGEST CHANGE             - E13.8 /             PSOR
     X                  11X 26HMAXIMUM ALLOWABLE CHANGE - E13.8 /               PSOR
     X                  11X 26HOVERRELAXATION PARAMETER - E13.8 //              PSOR
     X 1X 131(1H*))                                                             PSOR
C                                                                               PSOR
      CALL GENR(A,1,0,'0COEFFICIENTS OF P(J)')                                  PSOR
      DO 670 J=1,M                                                              PSOR
      K = J                                                                     PSOR
      CALL GENI(K,1,1,'0   COLUMN')                                             PSOR
      CALL GENR(A(1,J),1,M,' ')                                                 PSOR
  670 CONTINUE                                                                  PSOR
      CALL GENR(R,1,M,'0RIGHT HAND SIDE')                                       PSOR
      CALL GENR(X,1,M,'0COMPUTED VALUES OF P AFTER MAXIMUM ITERATIONS')         PSOR
      IF(BIGC-EPS) 700,700                                                      PSOR
C                                                                               PSOR
      RETURN                                                                    PSOR
C                                                                               PSOR
  700 CONTINUE                                                                  PSOR
  710 FORMAT(///7X' PSOR CONVERGED TO A SOLUTION FOR PRESSURES IN'I5,           PSOR
     X' ITERATIONS')                                                            PSOR
      RETURN 1                                                                  PSOR
      END                                                                       PSOR
```

```
      SUBROUTINE RADIR(NLOC,SIGMA,TZERO)
C
C  CALCULATION FOR IR CROSS RADIATION
C
      LOGICAL EXPLCT
      COMMON /SOURCE/ Q(1)
      COMMON /XSPACE/ NDIM, NTH, EXT(1)
      COMMON /ARRAY / CURVO(1)
      COMMON / DIMENS/ NON,NAN
      COMMON /TEMP  / T(1)
      COMMON /FIXCON/ CON(1)
C
      DIMENSION NLOC(1)
      DIMENSION NCURVO(1)
C
      EQUIVALENCE (CON(1),TIME), (CON(22),DTIMEI)
      EQUIVALENCE (CURVO,NCURVO)
C
      DEFINE DTAU(I) = EXT(NNC+I)
C
      NNT = NON + NAN
      NNC = NTH - NNT
      EXPLCT = .TRUE.
      IF(DTIMEI .GT. 0.0) EXPLCT = .FALSE.

      IF(NLOC(1) .EQ. 6) GO TO 2
      CALL TOPLIN
      WRITE(6,1) NLOC(1)
    1 FORMAT(57H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO RADIR, IC
     1= I5, 7H  * * *)
      CALL WLRBCK
      CALL EXIT
C
    2 ISNA  = NLOC(2)
      ISEM  = NLOC(3)
      ISREF = NLOC(4)
      ISCON = NLOC(5)
      NNA   = NLOC(6)
      ISEA  = NLOC(7)
      NS = NCURVO(ISNA+1)
      NC = NCURVO(ISCON)
      IF(NCURVO(ISEM))150,500
      IBEG = ISNA + 1
      IEND = ISNA + NCURVO(ISNA)
      CALL TOPLIN
      WRITE(6,400) (NCURVO(KK),KK=IBEG,IEND)
  400 FORMAT(   'OIR CROSS RADIATION DATA'///6X'SURFACE DATA'//11X'NUMBER
     X OF SURFACES ='I5//11X'SURFACE NUMBER'7X'SURFACE AREA'7X'NUMBER OF
     X NODES'//(20X15,7XF12.5,16X15))
      IF(NS .NE. (NCURVO(ISNA)-1)/3) GO TO 501
      WRITE(6,401) (CURVO(ISEM+KK),KK=1,NS)
  401 FORMAT(///6X'SURFACE EMISSIVITY DATA'//(12X10E12.5))
      IF(NS .NE. NCURVO(ISEM)) GO TO 500
      DO 20 I=1,NS
      LOC = ISEM + I
      IF (CURVO(LOC) .GT. 0.0) GO TO 10
      CURVO(LOC) = .00001
      GO TO 20
   10 IF (CURVO(LOC) .LT. 1.0) GO TO 20
```

```
      CURVO(LOC) = .99999                                                RADIR
   20 CONTINUE                                                           RADIR
      WRITE(6,403) (CURVO(ISREF+KK),KK=1,NS)                             RADIR
  403 FORMAT(///6X'SURFACE REFLECTIVITY DATA'//(12X10E12.5))             RADIR
      IF(NS .NE. NCURVO(ISREF)) GO TO 502                                RADIR
      DO 40 I=1,NS                                                       RADIR
      LOC = ISREF + I                                                    RADIR
      IF (CURVO(LOC) .GT. 0.0)GO TO 30                                   RADIR
      CURVO(LOC) = .00001                                                RADIR
      GO TO 40                                                           RADIR
   30 IF (CURVO(LOC) .LT. 1.0) GO TO 40                                  RADIR
      CURVO(LOC) = .99999                                                RADIR
   40 CONTINUE                                                           RADIR
      IBEG = ISCON + 1                                                   RADIR
      IEND = ISCON + NCURVO(ISCON)                                       RADIR
      WRITE(6,405) (NCURVO(KK),KK=IBEG,IEND)                             RADIR
  405 FORMAT(///6X'SURFACE CONNECTION DATA'//11X'FROM SURFACE'7X'TO SURF RADIR
     *ACE'8X'VIEW FACTOR'//(18X15,12X15,7XE12.5))                        RADIR
      LOC = NNA                                                          RADIR
      WRITE(6,410)                                                       RADIR
  410 FORMAT(///6X'NODE DATA'//11X'SURFACE'6X4('NODE'11X'AREA'6X)//)     RADIR
      DO 60 I=1,NS                                                       RADIR
      LL = ISNA + 3*I                                                    RADIR
      IBEG = LOC + 1                                                     RADIR
      IEND = LOC + NCURVO(LOC)                                           RADIR
      WRITE(6,415) NCURVO(LL-1), (NCURVO(KK),KK=IBEG,IEND)               RADIR
  415 FORMAT(13X15,(T19,4(I10,F15.5)))                                   RADIR
      IF(NCURVO(LL+1) .NE. NCURVO(LOC)/2) GO TO 504                      RADIR
      IST = LOC + 2                                                      RADIR
      LOC = LOC + NCURVO(LOC)                                            RADIR
      ASUM = 0.                                                          RADIR
      DO 50 J=IST,LOC,2                                                  RADIR
      ASUM = ASUM +  CURVO(J)                                            RADIR
   50 CONTINUE                                                           RADIR
      IF(ABS(ASUM-CURVO(LL))/CURVO(LL) .GT. .01) GO TO 505               RADIR
      LOC = LOC + 1                                                      RADIR
   60 CONTINUE                                                           RADIR
      IF( .5*(NS*NS+NS) .GT. NCURVO(ISEA)) GO TO 506                     RADIR
      LL = ISNA - 1                                                      RADIR
      LOC = ISCON                                                        RADIR
      DO 80 I=1,NC,3                                                     RADIR
      DO 75 J=1,2                                                        RADIR
      LOC = LOC + 1                                                      RADIR
      DO 70 K=1,NS                                                       RADIR
      IF(NCURVO(LOC) .EQ. NCURVO(LL+3*K)) GO TO 72                       RADIR
   70 CONTINUE                                                           RADIR
      GO TO 507                                                          RADIR
   72 NCURVO(LOC) = K                                                    RADIR
   75 CONTINUE                                                           RADIR
      LOC = LOC + 1                                                      RADIR
   80 CONTINUE                                                           RADIR
      DO 100 I=1,NC,3                                                    RADIR
      LOC = ISCON + I                                                    RADIR
      NFR = NCURVO(LOC)                                                  RADIR
      NTO = NCURVO(LOC+1)                                                RADIR
      LOC1 = ISNA + 3*NFR                                                RADIR
      EA = -CURVO(LOC+2)*CURVO(LOC1)                                     RADIR
      IF(NFR-NTO)95,95                                                   RADIR
      NHLD = NFR                                                         RADIR
      NFR  = NTO                                                         RADIR
      NTO = NHLD                                                         RADIR
   95 LOC2 = (NFR-1)*NS - (NFR*NFR-NFR)/2 + NTO                          RADIR
      CURVO(ISEA+LOC2) = EA                                              RADIR
```

```
  100 CONTINUE                                                               RADIR
      DO 110 I = 1,NS                                                        RADIR
      LOC = (I-1)*NS - (I*I-I)/2 + 1                                         RADIR
      CURVO(ISEA+LOC) = CURVO(ISEA+LOC) + CURVO(ISNA+3*I)/CURVO(ISREF+I)     RADIR
  110 CONTINUE                                                               RADIR
      CALL SYMINV(CURVO(ISEA+1),NS)                                          RADIR
      WRITE(6,417)                                                           RADIR
  417 FORMAT(///6X'CONNECTION DATA'//11X'FROM SURFACE'5X'TO SURFACE' 6X      RADIR
     X'SCRIPT FA'//)                                                         RADIR
      DO 120 I=1,NC,3                                                        RADIR
      LOC1 = ISCON + I                                                       RADIR
      NFR = NCURVO(LOC1)                                                     RADIR
      NTO = NCURVO(LOC1+1)                                                   RADIR
      LOC = (NFR-1)*NS - (NFR*NFR-NFR)/2 + NTO                               RADIR
      CURVO(LOC1+2) = CURVO(ISEM+NFR)*CURVO(ISEM+NTO)*CURVO(ISNA+3*NFR)*     RADIR
     X                CURVO(ISNA+3*NTO)*CURVO(ISEA+LOC)/                     RADIR
     X                CURVO(ISREF+NFR)/CURVO(ISREF+NTO)                      RADIR
      WRITE(6,420) NCURVO(ISNA+3*NFR-1), NCURVO(ISNA+3*NTO-1),               RADIR
     X             CURVO(LOC1+2)                                             RADIR
  420 FORMAT(12XI10,5XI10,E15.5)                                             RADIR
  120 CONTINUE                                                               RADIR
      DO 125 I=1,NS                                                          RADIR
      LOC1 = ISEA + NS * I - I                                               RADIR
      CURVO(LOC1) = 0.0                                                      RADIR
  125 CONTINUE                                                               RADIR
      DO 130 I = 1,NC,3                                                      RADIR
      LOC = ISCON + I                                                        RADIR
      LOC1 = ISEA + NCURVO(LOC) * NS - I                                     RADIR
      CURVO(LOC1) = CURVO(LOC1) + CURVO(LOC+2)                               RADIR
      LOC1 = ISEA + NCURVO(LOC+1) * NS - I                                   RADIR
      CURVO(LOC1) = CURVO(LOC1) + CURVO(LOC+2)                               RADIR
  130 CONTINUE                                                               RADIR
      NCURVO(ISEM) = -NCURVO(ISEM)                                           RADIR
      CALL TOPLIN                                                            RADIR
  150 LOC = NNA                                                              RADIR
      DO 175 I=1,NS                                                          RADIR
      LOC1 = ISNA + 3*I                                                      RADIR
      ISN  = NCURVO(LOC1-1)                                                  RADIR
      NN = NCURVO(LOC1+1)                                                    RADIR
      SUMAT = 0.0                                                            RADIR
      DO 170 J=1,NN                                                          RADIR
      LOC = LOC + 1                                                          RADIR
      NODE = NCURVO(LOC)                                                     RADIR
      TP = T(NODE) - TZERO                                                   RADIR
      LOC = LOC + 1                                                          RADIR
      SUMAT = SUMAT + CURVO(LOC)*TP*TP*TP*TP                                 RADIR
  170 CONTINUE                                                               RADIR
      T(ISN) = SUMAT/CURVO(LOC1)                                             RADIR
      LOC = LOC + 1                                                          RADIR
  175 CONTINUE                                                               RADIR
      DO 180 I=1,NS                                                          RADIR
      CURVO(ISEA+I-1) = 0.0                                                  RADIR
  180 CONTINUE                                                               RADIR
      DO 190 I = 1,NC,3                                                      RADIR
      LOC =ISCON + I                                                         RADIR
      LOC1 = LOC + 2                                                         RADIR
      LOC2 = ISEA + NCURVO(LOC) - 1                                          RADIR
      LOC3 = ISNA + 3*NCURVO(LOC+1) - 1                                      RADIR
      ISN = NCURVO(LOC3)                                                     RADIR
      CURVO(LOC2) = CURVO(LOC2) + SIGMA*CURVO(LOC1)*T(ISN)                   RADIR
      LOC2 = ISEA + NCURVO(LOC+1) - 1                                        RADIR
      LOC3 = ISNA + 3*NCURVO(LOC) - 1                                        RADIR
      ISN = NCURVO(LOC3)                                                     RADIR
```

```
      CURVO(LOC2) = CURVO(LOC2) + SIGMA*CURVO(LOC1)*T(ISN)                 RADIR
  190 CONTINUE                                                             RADIR
      LOC = NNA                                                            RADIR
      DO 210 I=1,NS                                                        RADIR
      LOC1 = ISNA + 3*I                                                    RADIR
      AREAS = CURVO(LOC1)                                                  RADIR
      NN = NCURVO(LOC1+1)                                                  RADIR
      LOC2 = ISEA + I - 1                                                  RADIR
      FA = CURVO(LOC2+NS)                                                  RADIR
      FAT4 = CURVO(LOC2)                                                   RADIR
      DO 200 J=1,NN                                                        RADIR
      LOC = LOC + 1                                                        RADIR
      NODE = NCURVO(LOC)                                                   RADIR
      LOC = LOC + 1                                                        RADIR
      ARAT = CURVO(LOC)/AREAS                                              RADIR
      TP = T(NODE) - TZERO                                                 RADIR
      SIGFAT = SIGMA*FA*TP*TP*TP                                           RADIR
      IF(EXPLCT) DTAU(NODE) = DTAU(NODE) + 4.0*SIGFAT*ARAT        *NEW     RADIR
      Q(NODE) = Q(NODE) + ARAT*(FAT4-TP*SIGFAT)                   **-1     RADIR
  200 CONTINUE                                                             RADIR
      LOC = LOC + 1                                                        RADIR
  210 CONTINUE                                                             RADIR
      DO 220 I = 1,NS                                                      RADIR
      ISN = NCURVO(ISNA+3*I-1)                                             RADIR
      T(ISN) = T(ISN)**.25 + TZERO                                         RADIR
  220 CONTINUE                                                             RADIR
      RETURN                                                               RADIR
  500 WRITE(6,600)                                                         RADIR
  600 FORMAT(1H0 131(1H*)//1X'THE SURFACE EMISSIVITY DATA DOES NOT HAVE    RADIR
     XTHE CORRECT NUMBER OF VALUES'//1X 131(1H*))                          RADIR
      GO TO 900                                                            RADIR
  501 WRITE(6,601)                                                         RADIR
  601 FORMAT(1H0 131(1H*)//1X'THE SURFACE DATA DOES NOT HAVE THE CORRECT   RADIR
     X NUMBER OF VALUES'//1X 131(1H*))                                     RADIR
      GO TO 900                                                            RADIR
  502 WRITE(6,602)                                                         RADIR
  602 FORMAT(1H0 131(1H*)//1X'THE SURFACE REFLECTIVITY DATA DOES NOT HAV   RADIR
     XE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                        RADIR
      GO TO 900                                                            RADIR
  504 WRITE(6,604) NCURVO(LL-1)                                            RADIR
  604 FORMAT(1H0 131(1H*)//1X'THE NODE DATA FOR SURFACE 'I5,' DOES NOT H   RADIR
     XAVE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                      RADIR
      GO TO 900                                                            RADIR
  505 WRITE(6,605) NCURVO(LL-1)                                            RADIR
  605 FORMAT(1H0 131(1H*)//1X'THE SUM OF THE AREAS FOR THE NODES ON SURF   RADIR
     XACE 'I5,' IS NOT EQUAL TO THE AREA OF THAT SURFACE'//1X 131(1H*))    RADIR
      GO TO 900                                                            RADIR
  506 WRITE(6,606)                                                         RADIR
  606 FORMAT(1H0 131(1H*)//1X'THE EXTRA SPACE ARRAY DOES NOT HAVE THE CO   RADIR
     XRRECT NUMBER OF SPACES'//1X 131(1H*))                                RADIR
      GO TO 900                                                            RADIR
  507 WRITE(6,607) NCURVO(LOC)                                             RADIR
  607 FORMAT(1H0 131(1H*)//1X'SURFACE NUMBER 'I5,' IN THE SURFACE CONNEC   RADIR
     XTION DATA WAS NOT SUPPLIED IN THE SURFACE DATA'//1X 131(1H*))        RADIR
  900 WRITE (6,910)                                                        RADIR
  910 FORMAT('0 ERROR OCCURED IN SUBROUTINE RADIR.'/                       RADIR
     X  ' EXECUTION TERMINATED BY A PROGRAMMED HALT.')                     RADIR
      CALL WLKBCK                                                          RADIR
      CALL EXIT                                                            RADIR
      END                                                                  RADIR
```

```
      SUBROUTINE RADSOL(NLOC)                                                  RADSOL
C  CALCULATION FOR SOLAR CROSS RADIATION                                       RADSOL
      COMMON /ARRAY / CURVO(1)                                                 RADSOL
      COMMON /FIXCON/ TIME                                                     RADSOL
      COMMON /SOURCE/ Q(1)                                                     RADSOL
C                                                                              RADSOL
      DIMENSION NLOC(1)                                                        RADSOL
      DIMENSION NCURVO(1), FMT(12)                                             RADSOL
      EQUIVALENCE (CURVO,NCURVO)                                               RADSOL
C                                                                              RADSOL
      IF(NLOC(1) .EQ. 7) GO TO 2                                               RADSOL
      CALL TOPLIN                                                              RADSOL
      WRITE(6,1) NLOC(1)                                                       RADSOL
    1 FORMAT(58H0* * *  INCORRECT NUMBER OF ELEMENTS INPUT TO RADSOL, IC       RADSOL
     1 = I5, TH  * * *)                                                        RADSOL
      CALL WLKBCK                                                              RADSOL
      CALL EXIT                                                                RADSOL
C                                                                              RADSOL
    2 ISNA  = NLOC(2)                                                  *NEW    RADSOL
      ISALP = NLOC(3)                                                  **-1    RADSOL
      ISREF = NLOC(4)                                                          RADSOL
      ISHT  = NLOC(5)                                                          RADSOL
      ISCON = NLOC(6)                                                          RADSOL
      NNA   = NLOC(7)                                                          RADSOL
      ISEA  = NLOC(8)                                                          RADSOL
      NS = NCURVO(ISNA+1)                                                      RADSOL
      NC = NCURVO(ISCON)                                                       RADSOL
      IF(NCURVO(ISALP))150,500                                                 RADSOL
      IBEG = ISNA + 1                                                          RADSOL
      IEND = ISNA + NCURVO(ISNA)                                               RADSOL
      CALL TOPLIN                                                              RADSOL
      WRITE(6,400) (NCURVO(KK),KK=IBEG,IEND)                                   RADSOL
  400 FORMAT(  '0SOLAR CROSS RADIATION DATA'///6X'SURFACE DATA'//11X'NUM       RADSOL
     XBER OF SURFACES ='I5//11X'SURFACE NUMBER'7X'SURFACE AREA'7X'NUMBER       RADSOL
     X OF NODES'//(20XI5,7XF12.5,16XI5))                                       RADSOL
      IF(NS .NE. (NCURVO(ISNA)-1)/3) GO TO 501                                 RADSOL
      WRITE(6,401) (CURVO(ISALP+KK),KK=1,NS)                                   RADSOL
  401 FORMAT(///6X'SURFACE ABSORPTIVITY DATA'//(12X10E12.5))                   RADSOL
      IF(NS .NE. NCURVO(ISALP)) GO TO 500                                      RADSOL
      DO 20 I=1,NS                                                             RADSOL
      LOC = ISALP + I                                                          RADSOL
      IF (CURVO(LOC) .GT. 0.0) GO TO 10                                        RADSOL
      CURVO(LOC) = .00001                                                      RADSOL
      GO TO 20                                                                 RADSOL
   10 IF (CURVO(LOC) .LT. 1.0) GO TO 20                                        RADSOL
      CURVO(LOC) = .99999                                                      RADSOL
   20 CONTINUE                                                                 RADSOL
      WRITE(6,403) (CURVO(ISREF+KK),KK=1,NS)                                   RADSOL
  403 FORMAT(///6X'SURFACE REFLECTIVITY DATA'//(12X10E12.5))                   RADSOL
      IF(NS .NE. NCURVO(ISREF)) GO TO 502                                      RADSOL
      DO 40 I=1,NS                                                             RADSOL
      LOC = ISREF + I                                                          RADSOL
      IF (CURVO(LOC) .GT. 0.0)GO TO 30                                         RADSOL
      CURVO(LOC) = .00001                                                      RADSOL
      GO TO 40                                                                 RADSOL
   30 IF (CURVO(LOC) .LT. 1.0) GO TO 40                                        RADSOL
      CURVO(LOC) = .99999                                                      RADSOL
   40 CONTINUE                                                                 RADSOL
      WRITE(6,404)                                                             RADSOL
```

```
  404 FORMAT(///6X'SURFACE INCIDENT HEAT DATA'//)                        RADSOL
      FMT(1) = 4H(12X                                                    RADSOL
      FMT(12) = 1H)                                                      RADSOL
      IBEG = ISHT + 1                                                    RADSOL
      IEND = ISHT + NCURVO(ISHT)                                         RADSOL
      IB = ISHT                                                          RADSOL
      IF(NCURVO(ISHT) .LT. 10) GO TO 46                                  RADSOL
      NUM = NCURVO(ISHT)/10*10                                           RADSOL
      DO 45 K=1,NUM,10                                                   RADSOL
      DO 44 M=1,10                                                       RADSOL
      IB = IB + 1                                                        RADSOL
      IF(IABS(NCURVO(IB)).LT. 99999  .AND. IABS(NCURVO(IB)).GT. 0)       RADSOL
     X GO TO 43                                                          RADSOL
      FMT(M+1) = 6HE12.5,                                                RADSOL
      GO TO 44                                                           RADSOL
   43 FMT(M+1) = 5HI9,3X                                                 RADSOL
   44 CONTINUE                                                           RADSOL
      WRITE(6,FMT) (CURVO(KK),KK=IBEG,IB)                                RADSOL
      IBEG = IBEG + 10                                                   RADSOL
   45 CONTINUE                                                           RADSOL
      IF(IBEG-1 .GE. IEND) GO TO 49                                      RADSOL
   46 J = 1                                                              RADSOL
      DO 47 M=IBEG,IEND                                                  RADSOL
      J = J + 1                                                          RADSOL
      IF(IABS(NCURVO(M)).LT. 99999  .AND. IABS(NCURVO(M)).GT. 0)         RADSOL
     X  GO TO 42                                                         RADSOL
      FMT(J) = 6HE12.5,                                                  RADSOL
      GO TO 47                                                           RADSOL
   42 FMT(J) = 5HI9,3X                                                   RADSOL
   47 CONTINUE                                                           RADSOL
      J = J + 1                                                          RADSOL
      DO 48 I=J,11                                                       RADSOL
      FMT(I) = 1H                                                        RADSOL
   48 CONTINUE                                                           RADSOL
      WRITE(6,FMT) (CURVO(KK),KK=IBEG,IEND)                              RADSOL
   49 IF(NS .NE. NCURVO(ISHT)) GO TO 503                                 RADSOL
      IBEG = ISCON + 1                                                   RADSOL
      IEND = ISCON + NCURVO(ISCON)                                       RADSOL
      WRITE(6,405) (NCURVO(KK),KK=IBEG,IEND)                             RADSOL
  405 FORMAT(///6X'SURFACE CONNECTION DATA'//11X'FROM SURFACE'7X'TO SURF RADSOL
     XACE'8X'VIEW FACTOR'//(18XI5,12XI5,7XE12.5))                        RADSOL
      LOC = NNA                                                          RADSOL
      WRITE(6,410)                                                       RADSOL
  410 FORMAT(///6X'NODE DATA'//11X'SURFACE'6X4('NODE'11X'AREA'6X)//)     RADSOL
      DO 60 I=1,NS                                                       RADSOL
      LL = ISNA + 3*I                                                    RADSOL
      IBEG = LOC + 1                                                     RADSOL
      IEND = LOC + NCURVO(LOC)                                           RADSOL
      WRITE(6,415) NCURVO(LL-1), (NCURVO(KK),KK=IBEG,IEND)               RADSOL
  415 FORMAT(13XI5,(T19,4(I10,F15.5)))                                   RADSOL
      IF(NCURVO(LL+1) .NE. NCURVO(LOC)/2) GO TO 504                      RADSOL
      IST = LOC + 2                                                      RADSOL
      LOC = LOC + NCURVO(LOC)                                            RADSOL
      ASUM = 0.                                                          RADSOL
      DO 50 J=IST,LOC,2                                                  RADSOL
      ASUM = ASUM +  CURVO(J)                                            RADSOL
   50 CONTINUE                                                           RADSOL
      IF(ABS(ASUM-CURVO(LL))/CURVO(LL) .GT. .01) GO TO 505               RADSOL
      LOC = LOC + 1                                                      RADSOL
   60 CONTINUE                                                           RADSOL
      IF(.5*(NS*NS+NS) .GT. NCURVO(ISEA)) GO TO 506                      RADSOL
      LL = ISNA - 1                                                      RADSOL
      LOC = ISCON                                                        RADSOL
```

```
      DO 80 I=1,NC,3                                                    RADSOL
      DO 75 J=1,2                                                       RADSOL
      LOC = LOC + 1                                                     RADSOL
      DO 70 K=1,NS                                                      RADSOL
      IF(NCURVO(LOC) .EQ. NCURVO(LL+3*K)) GO TO 72                      RADSOL
   70 CONTINUE                                                          RADSOL
      GO TO 507                                                         RADSOL
   72 NCURVO(LOC) = K                                                   RADSOL
   75 CONTINUE                                                          RADSOL
      LOC = LOC + 1                                                     RADSOL
   80 CONTINUE                                                          RADSOL
      DO 100 I=1,NC,3                                                   RADSOL
      LOC = ISCON + I                                                   RADSOL
      NFR = NCURVO(LOC)                                                 RADSOL
      NTO = NCURVO(LOC+1)                                               RADSOL
      LOC1 = ISNA + 3*NFR                                               RADSOL
      EA = -CURVO(LOC+2)*CURVO(LOC1)                                    RADSOL
      IF(NFR-NTO)95,95                                                  RADSOL
      NHLD = NFR                                                        RADSOL
      NFR  = NTO                                                        RADSOL
      NTO = NHLD                                                        RADSOL
   95 LOC2 = (NFR-1)*NS - (NFR*NFR-NFR)/2 + NTO                         RADSOL
      CURVO(ISEA+LOC2) = EA                                             RADSOL
  100 CONTINUE                                                          RADSOL
      DO 110 I = 1,NS                                                   RADSOL
      LOC = (I-1)*NS - (I*I-I)/2 + I                                    RADSOL
      CURVO(ISEA+LOC) = CURVO(ISEA+LOC) + CURVO(ISNA+3*I)/CURVO(ISREF+I)RADSOL
  110 CONTINUE                                                          RADSOL
      CALL SYMINV(CURVO(ISEA+1),NS)                                     RADSOL
      WRITE(6,417)                                                      RADSOL
  417 FORMAT(///6X'CONNECTION DATA'//11X'FROM SURFACE'5X'TO SURFACE' 6X RADSOL
     X'SCRIPT FA'//)                                                    RADSOL
      DO 120 I=1,NC,3                                                   RADSOL
      LOC1 = ISCON + I                                                  RADSOL
      NFR = NCURVO(LOC1)                                                RADSOL
      NTO = NCURVO(LOC1+1)                                              RADSOL
      LOC = (NFR-1)*NS - (NFR*NFR-NFR)/2 + NTO                          RADSOL
      CURVO(LOC1+2) = CURVO(ISEA+LOC)*CURVO(ISALP+NFR)*CURVO(ISNA+3*NFR)RADSOL
     X                *CURVO(ISNA+3*NTO)/CURVO(ISREF+NFR)               RADSOL
      WRITE(6,420) NCURVO(ISNA+3*NFR-1), NCURVO(ISNA+3*NTO-1),          RADSOL
     X              CURVO(LOC1+2)                                       RADSOL
  420 FORMAT(12XI10,5XI10,E15.5)                                        RADSOL
  120 CONTINUE                                                          RADSOL
      NCURVO(ISALP) = -NCURVO(ISALP)                                    RADSOL
      CALL TOPLIN                                                       RADSOL
  150 DO 160 I=1,NS                                                     RADSOL
      LOC = ISEA + I - 1                                                RADSOL
      CURVO(LOC) = 0.0                                                  RADSOL
      LOC = LOC + NS                                                    RADSOL
      LOC1 = ISHT + I                                                   RADSOL
      CURVO(LOC) = CURVO(LOC1)                                          RADSOL
      IF(IABS(NCURVO(LOC1)).LE. 99999  .AND. IABS(NCURVO(LOC1)).GT. 0)  RADSOL
     X  CURVO(LOC) = POL(NCURVO(LOC1),TIME)                             RADSOL
  160 CONTINUE                                                          RADSOL
      DO 180 I=1,NC,3                                                   RADSOL
      LOC = ISCON + I                                                   RADSOL
      NFR = NCURVO(LOC)                                                 RADSOL
      NTO = NCURVO(LOC+1)                                               RADSOL
      FA = CURVO(LOC+2)                                                 RADSOL
      LOC1 = ISEA + NFR - 1                                             RADSOL
      LOC2 = ISEA + NS - 1                                              RADSOL
      CURVO(LOC1) = CURVO(LOC1) + FA*CURVO(LOC2+NTO)                    RADSOL
      LOC1 = ISEA + NTO - 1                                             RADSOL
```

```
      CURVO(LOC1) = CURVO(LOC1) + FA*CURVO(LOC2+NFR)                      RADSOL
  180 CONTINUE                                                            RADSOL
      LOC = NNA                                                           RADSOL
      DO 220 I=1,NS                                                       RADSOL
      LOC1 = ISNA + 3*I                                                   RADSOL
      AREAS = CURVO(LOC1)                                                 RADSOL
      NN = NCURVO(LOC1+1)                                                 RADSOL
      SQ = CURVO(ISEA+I-1)                                                RADSOL
      DO 210 J=1,NN                                                       RADSOL
      LOC = LOC + 1                                                       RADSOL
      NODE = NCURVO(LOC)                                                  RADSOL
      LOC = LOC + 1                                                       RADSOL
      Q(NODE) = Q(NODE) + CURVO(LOC)/AREAS*SQ                             RADSOL
  210 CONTINUE                                                            RADSOL
      LOC = LOC + 1                                                       RADSOL
  220 CONTINUE                                                            RADSOL
      RETURN                                                              RADSOL
  500 WRITE(6,600)                                                        RADSOL
  600 FORMAT(1H0 131(1H*)//1X'THE SURFACE ABSORPTIVITY DATA DOES NOT HAV  RADSOL
     XE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                       RADSOL
      GO TO 900                                                           RADSOL
  501 WRITE(6,601)                                                        RADSOL
  601 FORMAT(1H0 131(1H*)//1X'THE SURFACE DATA DOES NOT HAVE THE CORRECT  RADSOL
     X NUMBER OF VALUES'//1X 131(1H*))                                    RADSOL
      GO TO 900                                                           RADSOL
  502 WRITE(6,602)                                                        RADSOL
  602 FORMAT(1H0 131(1H*)//1X'THE SURFACE REFLECTIVITY DATA DOES NOT HAV  RADSOL
     XE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                       RADSOL
      GO TO 900                                                           RADSOL
  503 WRITE(6,603)                                                        RADSOL
  603 FORMAT(1H0 131(1H*)//1X'THE SURFACE INCIDENT HEAT DATA DOES NOT HA  RADSOL
     XVE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                      RADSOL
      GO TO 900                                                           RADSOL
  504 WRITE(6,604) NCURVO(LL-1)                                           RADSOL
  604 FORMAT(1H0 131(1H*)//1X'THE NODE DATA FOR SURFACE 'I5,' DOES NOT H  RADSOL
     XAVE THE CORRECT NUMBER OF VALUES'//1X 131(1H*))                     RADSOL
      GO TO 900                                                           RADSOL
  505 WRITE(6,605) NCURVO(LL-1)                                           RADSOL
  605 FORMAT(1H0 131(1H*)//1X'THE SUM OF THE AREAS FOR THE NODES ON SURF  RADSOL
     XACE 'I5,' IS NOT EQUAL TO THE AREA OF THAT SURFACE'//1X 131(1H*))   RADSOL
      GO TO 900                                                           RADSOL
  506 WRITE(6,606)                                                        RADSOL
  606 FORMAT(1H0 131(1H*)//1X'THE EXTRA SPACE ARRAY DOES NOT HAVE THE CO  RADSOL
     XRRECT NUMBER OF SPACES'//1X 131(1H*))                               RADSOL
      GO TO 900                                                           RADSOL
  507 WRITE(6,607) NCURVO(LOC)                                            RADSOL
  607 FORMAT(1H0 131(1H*)//1X'SURFACE NUMBER 'I5,' IN THE SURFACE CONNEC  RADSOL
     XTION DATA WAS NOT SUPPLIED IN THE SURFACE DATA'//1X 131(1H*))       RADSOL
  900 WRITE (6,910)                                                       RADSOL
  910 FORMAT('0 ERROR OCCURED IN SUBROUTINE RADSOL.'/                     RADSOL
     X ' EXECUTION TERMINATED BY A PROGRAMMED HALT.')                     RADSOL
      CALL WLKBCK                                                         RADSOL
      CALL EXIT                                                           RADSOL
      END                                                                 RADSOL
```

```
      SUBROUTINE REVPOL(Y,A,X)                                          REVPOL
C                                                                       REVPOL
      DIMENSION A(1)                                                    REVPOL
C                                                                       REVPOL
      EQUIVALENCE (D,N)                                                 REVPOL
C                                                                       REVPOL
C                                                                       REVPOL
      D = A(1)                                                          REVPOL
      N = N                                                             REVPOL
      IF(MOD(N,2) .GT. 0) GO TO 20                                      REVPOL
      IF(A(N+1) .GT. A(3)) GO TO 16                                     REVPOL
      X = A(2)                                                          REVPOL
      IF(Y .GE. A(3)) RETURN                                            REVPOL
      X = A(N)                                                          REVPOL
     .IF(Y .LE. A(N+1)) RETURN                                          REVPOL
      DO 15 I=4,N,2                                                     REVPOL
      IF(Y - A(I+1)) 15,10,5                                            REVPOL
    5 X = A(I-2) + (Y-A(I-1))*(A(I)-A(I-2))/(A(I+1)-A(I-1))             REVPOL
      RETURN                                                            REVPOL
   10 X = A(I)                                                          REVPOL
      RETURN                                                            REVPOL
   15 CONTINUE                                                          REVPOL
      GO TO 20                                                          REVPOL
   16 X = A(2)                                                          REVPOL
      IF(Y .LE. A(3)) RETURN                                            REVPOL
      X = A(N)                                                          REVPOL
      IF(Y .GE. A(N+1)) RETURN                                          REVPOL
      DO 19 I=4,N,2                                                     REVPOL
      IF(Y - A(I+1)) 17,18,19                                           REVPOL
   17 X = A(I-2) + (Y-A(I-1))*(A(I)-A(I-2))/(A(I+1)-A(I-1))             REVPOL
      RETURN                                                            REVPOL
   18 X = A(I)                                                          REVPOL
      RETURN                                                            REVPOL
   19 CONTINUE                                                          REVPOL
   20 WRITE(6,25) A(1)                                                  REVPOL
   25 FORMAT(36H WRONG ARRAY LENGTH FOR REVPOL, IC = I5)                REVPOL
      CALL WLKBCK                                                       REVPOL
      CALL EXIT                                                         REVPOL
      END                                                               REVPOL
```

```
      SUBROUTINE SYMSOL(A,N,B,*)                                        SYMSOL
C                                                                       SYMSOL
      DIMENSION A(1), B(1)                                              SYMSOL
      DOUBLE PRECISION DIN, WORK                                        SYMSOL
C     FACTORIZE GIVEN MATRIX BY MEANS OF SUBROUTINE MFSD                SYMSOL
C     A = TRANSPOSE(T)*T                                                SYMSOL
      CALL MFSD(A,N,$15)                                                SYMSOL
C     INVERT UPPER TRIANGULAR MATRIX T/PREPARE INVERSION LOOP           SYMSOL
    1 IPIV = N*(N+1)/2                                                  SYMSOL
      IND = IPIV                                                        SYMSOL
C     INITIALIZE INVERSION-LOOP                                         SYMSOL
      DO 6 I=1,N                                                        SYMSOL
      DIN = 1.D0/A(IPIV)                                                SYMSOL
      A(IPIV) = DIN                                                     SYMSOL
      MIN = N                                                           SYMSOL
      KEND = I - 1                                                      SYMSOL
      LANF = N - KEND                                                   SYMSOL
      IF(KEND) 5,5,2                                                    SYMSOL
    2 J = IND                                                           SYMSOL
C     INITIALIZE ROW-LOOP                                               SYMSOL
      DO 4 K=1,KEND                                                     SYMSOL
      WORK = 0.D0                                                       SYMSOL
      MIN = MIN - 1                                                     SYMSOL
      LHOR = IPIV                                                       SYMSOL
      LVER = J                                                          SYMSOL
C     START INNER LOOP                                                  SYMSOL
      DO 3 L=LANF,MIN                                                   SYMSOL
      LVER = LVER + 1                                                   SYMSOL
      LHOR = LHOR + L                                                   SYMSOL
    3 WORK = WORK + A(LVER)*A(LHOR)                                     SYMSOL
      A(J) = -WORK*DIN                                                  SYMSOL
    4 J = J - MIN                                                       SYMSOL
    5 IPIV = IPIV - MIN                                                 SYMSOL
    6 IND = IND - 1                                                     SYMSOL
      LD = N*(N+1)/2                                                    SYMSOL
      DO 12 I=N,1,-1                                                    SYMSOL
      LD = LD - I                                                       SYMSOL
      SUM = 0.0                                                         SYMSOL
      DO 11 J=1,I                                                       SYMSOL
      SUM = SUM + A(LD+J)*B(J)                                          SYMSOL
   11 CONTINUE                                                          SYMSOL
      B(I) = SUM                                                        SYMSOL
   12 CONTINUE                                                          SYMSOL
      LD = 0                                                            SYMSOL
      DO 8 I=1,N                                                        SYMSOL
      LD = LD + I                                                       SYMSOL
      K = LD                                                            SYMSOL
      SUM = 0.0                                                         SYMSOL
      DO 7 J=I,N                                                        SYMSOL
      SUM = SUM + A(K)*B(J)                                             SYMSOL
      K = K + J                                                         SYMSOL
    7 CONTINUE                                                          SYMSOL
      B(I) = SUM                                                        SYMSOL
    8 CONTINUE                                                          SYMSOL
      RETURN                                                            SYMSOL
   15 RETURN 4                                                          SYMSOL
      END                                                               SYMSOL
```

```
                                                                            TIMCHK
      SUBROUTINE TIMCHK(RTIME,KODE)                                         TIMCHK
C                                                                   *NEW    TIMCHK
      COMMON / FIXCON / CON(1)                                      *NEW    TIMCHK
C                                                                   *NEW    TIMCHK
      EQUIVALENCE (CON(1),TIMEN), (CON(3),TIMEND), (CON(18),OUTPUT) **-1    TIMCHK
C                                                                           TIMCHK
      DATA CTIMEI / 0.0 /                                                   TIMCHK
C                                                                           TIMCHK
      IF(CTIMEI .GT. 0.0) GO TO 100                                         TIMCHK
      CALL CLOCK(CTIMEI)                                                    TIMCHK
      ETIME = 0.0                                                           TIMCHK
      GO TO 200                                                             TIMCHK
  100 CALL CLOCK(CTIME)                                                     TIMCHK
      ETIME = CTIME - CTIMEI                                                TIMCHK
  200 IF(KODE .EQ. 0) GO TO 350                                             TIMCHK
      CALL LINECK(3)                                                        TIMCHK
      WRITE(6,300) ETIME                                                    TIMCHK
  300 FORMAT(16HOCOMPUTER TIME = F9.3, 8H MINUTES)                          TIMCHK
  350 IF(ETIME .LT. RTIME) RETURN                                   **-1    TIMCHK
      CALL LINECK(2)                                                *NEW    TIMCHK
      IF(KODE .EQ. 0) GO TO 450                                             TIMCHK
      WRITE(6,400) RTIME                                                    TIMCHK
  400 FORMAT(67HOEXECUTION TERMINATED BECAUSE COMPUTER TIME EXCEEDS TIME    TIMCHK
     1 REQUESTED, F9.3, 8H MINUTES)                                 *NEW    TIMCHK
      GO TO 500                                                     *NEW    TIMCHK
  450 WRITE(6,475) RTIME                                            *NEW    TIMCHK
  475 FORMAT(38HOCOMPUTER TIME EXCEEDS TIME REQUESTED, F9.3, 8H MINUTES)    TIMCHK
  500 TIMEND = TIMEN                                                *NEW    TIMCHK
      OUTPUT = 0.0                                                          TIMCHK
      RETURN                                                                TIMCHK
      END
```

```
      SUBROUTINE TOPLIN                                                     TOPLIN
C                                                                           TOPLIN
      COMMON /TITLE / H(20)                                                 TOPLIN
      COMMON /FIXCON/ N(1)                                                  TOPLIN
C                                                                           TOPLIN
      IF(N(28) .EQ. 11) RETURN                                              TOPLIN
      N(28) = 11                                                            TOPLIN
      N(29) = N(29) + 1                                                     TOPLIN
      WRITE(6,100) N(29), H                                                 TOPLIN
  100 FORMAT(124HITRW SYSTEMS IMPROVED NUMERICAL DIFFERENCING ANALYZER      TOPLIN
     * - - - SINDA - - -  UNIVAC-1108 FORTRAN-V VERSION        PA           TOPLIN
     *GE  , I5 // 5X 20A6)                                        *NEW      TOPLIN
      RETURN                                                      **-1      TOPLIN
      END                                                                   TOPLIN
```

TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT TPRNT

```
      SUBROUTINE TPRNT                                                TPRNT
C                                                                     TPRNT
      LOGICAL LSRT, CHK                                               TPRNT
C                                                                     TPRNT
      DIMENSION EXT(1)                                                TPRNT
C                                                                     TPRNT
      COMMON /TEMP  / T(1)                                            TPRNT
      COMMON /XSPACE/ NDIM, NTH, NEXT(1)                              TPRNT
      COMMON /FIXCON/ KON(1)                                          TPRNT
      COMMON /DIMENS/ NND, NNA, NNT                                   TPRNT
      COMMON /POINTN/ LNODE                                           TPRNT
C                                                                     TPRNT
      EQUIVALENCE (NEXT,EXT)                                          TPRNT
C                                                                     TPRNT
      DATA LSRT / .FALSE. /                                           TPRNT
      DATA HT  / 1HT /                                                TPRNT
C                                                                     TPRNT
      IF(LNODE .EQ. 0) CALL NNREAD(1)                                 TPRNT
      CALL STNDRD                                                     TPRNT
      IF(LSRT) GO TO 50                                               TPRNT
      LSRT = .TRUE.                                                   TPRNT
      NDIM = NDIM - NNT                                               TPRNT
      IF(NDIM .LT. 0) GO TO 100                                       TPRNT
      NNODE = NDIM + NTH                                   *NEW       TPRNT
      DO 10 I=1,NNT                                        **-1       TPRNT
      NEXT(NNODE+I) = I                                               TPRNT
   10 CONTINUE                                                        TPRNT
      DO 30 J=2,NNT                                                   TPRNT
      K = NNT - J + 1                                                 TPRNT
      CHK = .TRUE.                                                    TPRNT
      DO 20 N=1,K                                                     TPRNT
      NN  = NEXT(NNODE+N  )                                           TPRNT
      NN1 = NEXT(NNODE+N+1)                                           TPRNT
      IF(NEXT(LNODE+NN) .LE. NEXT(LNODE+NN1)) GO TO 20                TPRNT
      CHK = .FALSE.                                                   TPRNT
      NEXT(NNODE+N  ) = NN1                                           TPRNT
      NEXT(NNODE+N+1) = NN                                            TPRNT
   20 CONTINUE                                                        TPRNT
      IF(CHK) GO TO 50                                                TPRNT
   30 CONTINUE                                                        TPRNT
   50 IF(NDIM .LT. 12) GO TO 100                                      TPRNT
      J = 1                                                           TPRNT
      L = 6                                                           TPRNT
      M = NTH + 1                                                     TPRNT
   60 IF(L .GT. NNT) L = NNT                                          TPRNT
      K = M                                                           TPRNT
      DO 70 I=J,L                                                     TPRNT
      N = NEXT(NNODE+I)                                               TPRNT
      NEXT(K) = NEXT(LNODE+N)                                         TPRNT
      EXT(K+1) = T(N)                                                 TPRNT
      K = K + 2                                                       TPRNT
   70 CONTINUE                                                        TPRNT
      K = K - 1                                                       TPRNT
      IF(KON(28) .LT. 60) GO TO 80                                    TPRNT
      CALL TOPLIN                                                     TPRNT
      WRITE(6,75)                                                     TPRNT
   75 FORMAT(1H )                                                     TPRNT
      KON(28) = KON(28) + 1                                           TPRNT
   80 WRITE(6,90) (HT, NEXT(I), EXT(I+1),I=M,K,2)                     TPRNT
```

```
   90 FORMAT(6(1X, A1, I6, 1H=, G12.5, 1X))                               TPRNT
      KON(28) = KON(28) + 1                                               TPRNT
      IF(L .EQ. NNT) RETURN                                               TPRNT
      J = L + 1                                                           TPRNT
      L = L + 6                                                           TPRNT
      GO TO 60                                                            TPRNT
  100 WRITE(6,110) NDIM                                                   TPRNT
  110 FORMAT(75H0* * *  INSUFFICIENT DYNAMIC STORAGE AVAILABLE FOR SUBRO  TPRNT
     1UTINE TPRNT, NDIM = 15, 7H  * * *)                                  TPRNT
      STOP                                                                TPRNT
      END                                                                 TPRNT
```

APPENDIX B

USERS DESCRIPTION FOR PLOTA

This appendix presents user descriptions for a SINDA plotting routine, PLOTA, and a tape combining routine, MCOMB. Both routines are on the second file of the SINDA/Version 9 program tape but are main routines rather than user subroutines. A brief description of the routines and the user input description is given below.

PLOTA DESCRIPTION

The plot routine which is available on the SINDA program tape can be used with a history tape from a previous SINDA run to generate microfilm output. The items available for plotting are (1) pressures for each pressure node or pressure drop values for each tube, (2) valve positions for each valve, (3) flowrates for each tube and (4) temperatures for each temperature lump. Each of these items may be plotted as a function of mission time. The user specifies the grid time range to be plotted, a time label, and the items to be plotted. A number of history tapes may be combined prior to plotting the results. The user has the option of averaging any portion of the plotted curve and of specifying the range of the ordinate axis.

The system control cards and the data input cards for PLOTA are described below:

SYSTEM CONTROL CARDS FOR PLOTA

$^{7}_{8}$Z RUN

$^{7}_{8}$N MSG

$^{7}_{8}$ PLT

$^{7}_{8}$ ASG A=XXX (SINDA PROGRAM TAPE)

$^{7}_{8}$ ASG E=XXX (FIRST TAPE TO BE COMBINED)

$^{7}_{8}$ ASG F=XXX (SECOND TAPE TO BE COMBINED)
Add additional ASG cards as required for
tapes to be combined

```
7
8   ASG T=XXX (COMBINED TAPE)
7
8   XQT CUR
    TRW A
    PEF A
    IN A
    TRI A
7
8   XQT PLOTA
    DATA CARDS
7
8   EØF
```

PLOTA DATA CARDS

Columns	Format	Title	Description
Card 1 (Title Card)			
1-72	12A6	TITLEA	Any 72 alphammeric characters to be used as heading for each frame of plots
Card 2 (Parameter Card)			
1-10	F10.0	TA	First value of time to be plotted (hours).
11-20	F10.0	TZ	Last value of time to be plotted (hours).
21-30	F10.0	TPG	Time range for each grid. Number of grids drawn will be (TZ-TA)/TPG. (If TPG is left blank, the job will terminate.)
31-35	I5	ITMX	Time scale lable: = 1, "SECONDS" = 2, "MINUTES" = 3, "HOURS" Any other value, "*****"
36-40	I5	MPNT	Print control code = 1, prints information to be plotted while loading the plot tape ≠ 1, will not print information to be plotted

Columns	Format	Title	Description
41-45	I5	NTP	Number of tapes to be combined. Use a negative number if start and/or stop times are specified on Card 3 for any tape to be combined.
46-50	I5	KT	Logical unit number to which tape to be plotted is assigned. If left blank, unit 23 is assumed. (See Table B-I) The combined tape is assigned to this unit.
51-55	I5	INC	= 1, every time point and associated data value from the tapes to be combined will be transferred to the combined tape. = 2, every second time point and associated data values will be transferred to the combined tape. etc.
56-60	I5	IUNIT	Logical unit number to which first tape to be combined is assigned. If left blank, unit 7 is assumed.
61-70	F10.0	ASTRT	Beginning time for averages (hours).
71-80	F10.0	ASTØP	Ending time for averages (hours).

Card 3 (Required only if NTP < 0. See Card 2 columns 41-45)

1-5	F5.3	XSTART	First time point from first tape to be combined which will be transferred to the combined tape.
6-10	F5.3	XSTØP	Last time point from first tape to be combined which will be transferred to the combined tape.

Repeat XSTART and XSTØP in five column fields for each tape to be combined.

Card 4 (Item Card)

1-5	I5	ITEM	The item number to be plotted. Use a negative value if this item is to start a

Columns	Format	Title	Description
			new grid. A maximum of four curves may be plotted on one grid. Insert a blank card when the number of items exceeds 34000 divided by the number of points between TA and TZ. More item cards may follow this blank card.
6-7	A2	ITYPE	A two character item type code which determines the type of item to be plotted. = XX, pressures or pressure drop values = VP, valve positions = FR, flow rates = ST, node temperatures
8	I1	IREL	= 0, node numbers are actual numbers = 1, node numbers are relative numbers
9-10	I2	KAVG	> 0, calculate the numerical average of this item over the interval specified by ASTRT and ASTØP in columns 61-70 and 71-80 of Card 2. > 9, plot the average on the frame with this item.
11-58	8A6	TITLES	Item description to be printed at the top of each grid, along with the plotting symbol which is generated and used by the program.
59-60			Blank

The next two values are optional (may be left blank) on the cards whose item numbers are negative and are ignored on all other item cards.

Columns	Format	Title	Description
61-70	F10.0	YLØ	The minimum (reference) value on the Y-axis
71-80	F10.0	YHI	The maximum value on the Y-axis

The above limits will be replaced by the program if numbers outside this range are found in the histories of the items to be plotted on this grid.

Columns	Format	Title	Description

Repeat Card 4 for each item to be plotted.

Card 5

1-80		Blank

Card 6

1-80		Blank

If additional history tapes are to be plotted, repeat Card 1 and subsequent cards for each additional history tape.

COMBINE ROUTINE DESCRIPTION

The combine routine, MCOMB, can be used to combine as many as six history tapes into one history tape prior to its being plotted or being compared to another tape. The combined tape which is generated can be saved for future use if required. The user selects the frequency with which the time points and associated data values on the original tapes are added to the new tape. That is, every time point on the original tape can be added to the new tape or every second, third, etc., point can be added depending on the requirements for the combined tape.

The combine routine is a very useful feature if several history tapes are generated on a long mission run. By combining these tapes before plotting, a continuous plot of the mission can be obtained. The convenience of the combine routine can also be observed when mission runs made with different time increments are compared. Obviously, the run made with the smaller time increment will take more computer time than the run made with the larger time increment, and will probably require at least one "restart". In such a situation, there would be two history tapes with the smaller time increment to compare to one with the larger time increment. Normally, this would take two separate runs. However, with the new arrangement, the two tapes with the smaller time increment can be combined and then compared to the tape with the larger time increment on the same run.

The system control cards and the data input cards for MCOMB are described below.

USER'S MANUAL FOR MCOMB ROUTINE

CONTROL AND PROGRAM CARDS

```
7
8Z   RUN

7
8N   MSG

7
8S   ASG  A=XXX (SINDA PROGRAM TAPE)

7
8    ASG  E=XXX (First Tape to be combined)

7
8    ASG  F=XXX (Second tape to be combined)
                Add additional ASG cards as required for
                tapes to be combined.

7
8    ASG  K=XXX (Combined Tape)

7
8    XQT  CUR

     TRW  A

     PEF  A

     IN  A

     TRI  A

7
8    XQT  MCOMB

     DATA CARDS

7
8    EØF
```

DATA CARDS FOR MCOMB ROUTINE

Columns	Format	Title	Description
Card 1 (Parameter Card)			
1-5	I5	NTAPE	Number of tapes to be combined. Use a negative value if start and/or stop times are specified on Card 2.
6-10	I5	IUNIT	Logical unit number to which first tape to be combined is assigned. If left blank unit 7 is assumed.
11-15	I5	KT	Logical unit number to which combined tape is assigned. If left blank, unit 13 is assumed.
16-20	I5	KODE2	= 1, time to be added to the times read from tapes to be combined will be supplied on Card 3. = 0, transfer times as read from tapes to be combined.
21-25	I5	INC	= 1, every time point and associated data values read from the tapes to be combined will be transferred to the combined tape. = 2, every second time point and associated data values will be transferred to the combined tape. etc.
Card 2 (Required only if NTAPE < 0. See Card 1 columns 1-5)			
1-5	F5.3	XSTART	First time point from first tape to be combined which will be transferred to the combined tape.
6-10	F5.3	XSTOP	Last time point from first tape which will be transferred to the combined tape.

Columns	Format	Title	Description

Repeat XSTART and XSTOP in five columns fields for each tape to be combined.

Card 3 (Required only if KØDE2 > 0. See Card 1 columns 16-20)

1-10	F10.0	ADD	Time to be added to each time read from first tape to be combined.

Repeat ADD in 10 column fields for each tape to be combined.

TABLE B-I

CORRESPONDENCE BETWEEN FORTRAN UNIT NO. & I/O DEVICE

FORTRAN UNIT NO.	I/O DEVICE
1	A
2	B
3	C
4	D
7	E
8	F
9	G
10	H
11	I
12	J
13	K
14	L
15	M
16	N
18	O
19	P
20	Q
21	R
22	S
23	T
24	U
25	V
26	W
27	X
28	Y
29	Z