Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

Brad J. Merrill

April 2007
Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

Brad J. Merrill

April 2007

Idaho National Laboratory
Idaho Falls, Idaho 83415

Prepared for the
U.S. Department of Energy
Office of Science Education and Technical Information
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517
Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

INL/EXT-07-12493

April 2007

Approved by

Brad J. Merrill/Author  4/6/07


Richard L. Moore/Peer Reviewer  4/10/07

Phil Sharpe/Approver/Manager  4/16/07
ABSTRACT

This report documents recent changes made to the MELCOR 1.8.2 computer code for application to the International Thermonuclear Experimental Reactor (ITER), as required by ITER Task Agreement ITA 81-18. There are four areas of change documented by this report. The first area is the addition to this code of a model for transporting HTO. The second area is the updating of the material oxidation correlations to match those specified in the ITER Safety Analysis Data List (SADL). The third area replaces a modification to an aerosol transport subroutine that specified the nominal aerosol density internally with one that now allows the user to specify this density through user input. The fourth area corrected an error that existed in an air condensation subroutine of previous versions of this modified MELCOR code. The appendices of this report contain FORTRAN listings of the coding for these modifications.
CONTENTS

ABSTRACT ........................................................................................................................................ iii

1. INTRODUCTION ....................................................................................................................... 1

2. HTO TRANSPORT MODEL ...................................................................................................... 2
   2.1 HTO Transport Modifications ............................................................................................. 2
   2.2 HTO Transport Modification Test Problems ....................................................................... 3

3. OXIDATION CORRELATION UPDATE .................................................................................. 8

4. CORRECTIONS TO THE AEROSOL TRANSPORT SUBROUTINE RN1RN4 ......................... 11

5. CORRECTIONS TO THE AIR CONDENSATION SUBROUTINES ........................................... 12

6. REFERENCES ............................................................................................................................. 15

Appendix A- Code Listing of changes to MELCOR for HTO Transport Model ....................... 16
Appendix B- Code Listing for Oxidation Modifications ............................................................... 31
Appendix C- Code Listing for Aerosol Transport Modifications ............................................... 50
Appendix D- Code Listing for Air Condensation Model Modifications .................................... 54

FIGURES

1. HTO pool and atmosphere masses for HTO transport test problem one ................................. 4
2. HTO pool and atmosphere masses for HTO transport test problem two ............................... 5
3. HTO pool and atmosphere masses for HTO transport test problem three ............................ 6
4. Total HTO pool and atmosphere masses for HTO transport test problem three ....................... 7

TABLES

1. Comparison of condensation model enthalpy change with values from Reference 10 for oxygen .... 14
2. Comparison of condensation model enthalpy change with values from Reference 10 for nitrogen .... 14
Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

1. INTRODUCTION

This report documents recent changes made to the MELCOR 1.8.2 computer code\(^1\) for application to the International Thermonuclear Experimental Reactor (ITER), as required by ITER Task Agreement ITA 81-18.\(^2\) There are four areas of change documented by this report. The first area is the addition to this code of a model for transporting tritiated water (HTO). The second area is the updating of the material oxidation correlations to match those specified in the latest version of the ITER Safety Analysis Data List (SADL).\(^3\) The third area replaces a modification to an aerosol transport subroutine that specified the nominal aerosol density internally with one that now allows the user to specify this density through user input. The fourth area corrected an error that existed in an air condensation subroutine of previous versions of this modified MELCOR code. The sections which follow describe these changes in more detail.
2. HTO TRANSPORT MODEL

A simple transport model for tritiated water (HTO) was added to a modified version the MELCOR 1.8.2 code for use in ITER safety analyses. The new version of MELCOR 1.8.2 was used for the ITER-FEAT Generic Site Safety Report (GSSR) analyses, but this particular feature was until now undocumented. This HTO transport model allows MELCOR to track tritium releases from components inside of the ITER vacuum vessel (VV) to the environment during accident conditions. A conservative assumption that has been adopted by ITER regarding tritium is that unless otherwise justified, any tritium gas mobilized will be immediately oxidized and be released in the oxide form (HTO), which is much more radiotoxic than the elemental tritium gas (T2). Once HTO is released into the atmosphere of a given enclosure, HTO will act like water by condensing on cooler surfaces such as a water pool or a building wall, evaporating from a hot water pool or wall, and moving between enclosures as a flowing gas. The following sub-sections of this report present the equations and assumptions on which this model is based, and present the results of three test problems that demonstrate that the model is working properly. Finally, because the intent of this report is to document modifications made to the MELCOR 1.8.2 code, Appendix A contains a FORTRAN listing of the changes made to MELCOR 1.8.2 for this HTO transport model.

2.1 HTO Transport Modifications

This HTO transport model accounts for HTO transport as a result of mass movement between volumes of a MELCOR model and between the pool and atmosphere of a given MELCOR volume by the mechanism of bulk boiling, bulk condensation, and surface condensation. To minimize the effort of implementing this model in MELCOR, the aerosol mass transport equations solved by MELCOR were adapted for use by this HTO transport model. As this version of MELCOR now operates, the first fifteen material classes remain the standard MELCOR classes, but if there are more than fifteen material classes, that is, the user has defined additional material classes, and if the highest material class is designated HTO, then the HTO transport model is activated. For the HTO material class, the aerosol agglomeration and deposition processes are bypassed since HTO is a gas or liquid and not aerosol. The resulting conservation of mass equations that constitute this HTO transport model can be written as follows:

\[
\frac{\partial M_{i_{HTO}}^a}{\partial t} = \dot{b} M_{i_{H2O}}^b x_{i_{HTO}}^p - M_{i_{H2O}}^b x_{i_{HTO}}^a + \sum_j A_{c_j} v_{j}^a M_{j_{HTO}}^a / V_j^a
\]

\[
+ \sum_i A_{c_i} v_{i}^a M_{i_{HTO}}^a / V_i^a - \sum_i A_{s_i} \Gamma_{i_{H2O}}^c x_{i_{HTO}}^a
\]

for the atmosphere of the “i”th control volume, where

\( M_{i_{HTO}} \) is the mass of HTO in the atmosphere (kg)
\( \dot{b} \) is the bulk pool H2O boiling rate (kg/s)
\( x_{i_{HTO}}^p \) is the HTO to H2O mass ratio in the pool
\( \dot{c} \) is the bulk atmosphere H2O condensation rate (kg/s)
\( x_{i_{HTO}}^a \) is the HTO to H2O mass ratio in the atmosphere
\( A_{i,j} \) are cross-sectional flow areas of flow paths between volumes \( i,j \) (m\(^2\))

\( v^a_{i,j} \) are atmosphere flow velocities of flow paths between volumes \( i,j \) (m/s)

\( V^a_{i,j} \) are atmosphere volumes in control volumes \( i,j \) (m\(^3\))

\( A_s \) is the surface area of the “lth” heat structure (m\(^2\))

\( \Gamma^c_{iH_2O} \) is the H\(_2\)O condensation flux on the “lth” heat structure (kg/m\(^2\)-s)

and, as:

\[
\frac{\partial M^p_{iHTO}}{\partial t} = -M^p_{iH_2O} x^p_{iHTO} + M^p_{iH_2O} x^a_{iHTO} + \sum_j A^c_{i,j} v^p_j M^p_{jHTO} / V^p_j - \sum_i A^c_{i,j} v^p_i M^p_{iHTO} / V^p_i + \sum_j A^c_{i,j} \Gamma^c_{iH_2O} x^a_{iHTO}
\]

(2)

for the pool of the “ith” control volume, where

\( M^p_{iHTO} \) is the mass of HTO in the pool (kg)

\( v^p_{i,j} \) are pool flow velocities of flow paths between volumes \( i,j \) (m/s)

\( V^p_{i,j} \) are pool volumes in control volumes \( i,j \) (m\(^3\)).

As is the case with all of the MELCOR aerosol material classes, this HTO transport model tracks the mass in all of the user defined aerosol bins, even though an aerosol size distribution has no meaning for the HTO material class. The input for this model is the standard MELCOR user input for a user defined material class. However, film retention of HTO is not considered by this model.

### 2.2 HTO Transport Modification Test Problems

Three problems were developed to test the operation of this HTO transport model. The first problem is a single control volume that contains water. Within the water pool of this control volume resides one kilogram of HTO. Energy inputs to the pool and atmosphere of this control volume are defined so that the pool boils dry, and then the atmosphere re-condensed within an arbitrary time span of 1000 seconds. This problem tests the movement of HTO between the liquid and vapor phases of water due to direct energy inputs into the water. Figure 1 contains the pool and atmosphere HTO masses for this test problem. As can be seen, total mass is being conserved, and the time histories of these quantities appear as a conjugate pair.

The second test problem adds a heat structure to the control volume of the first test problem. The boundary condition for the back of this heat structure is defined to boil the pool dry, and then re-condense the atmosphere within an arbitrary time span of 1000 seconds. This problem tests the movement of HTO between phases due to heat transfer driven mass transport processes. Figure 2 contains the pool and atmosphere HTO masses for this test problem. As can be seen, total mass is being conserved, and the time histories of these quantities appear as a conjugate pair.

The final test problem is a two control volume problem, where in the first control volume a pool energy source is used to boil off the liquid, and in the second control volume an atmosphere energy sink is used to re-condense this vapor. These control volumes are connected by a single flow path. This test
problem checks the transport of the HTO between control volumes. Figure 3 contains the pool and atmosphere HTO masses for volumes 1 and 2. The pool mass in volume 1 boils off in 250 seconds, the atmosphere HTO masses rise to about 0.5 kg as flow between volumes distributes the vaporized HTO, and then the pool HTO mass in volume 2 approaches 1.0 kg as the atmosphere in volume 2 re-condenses following 750 seconds. The total HTO mass in the atmosphere and pool appear in Figure 4. The trends in this figure are similar to the previous test problems. Based on the results of these test problems, the HTO transport model is working as intended.

Figure 1. HTO pool and atmosphere masses for HTO transport test problem one.
Figure 2. HTO pool and atmosphere masses for HTO transport test problem two.
Figure 3. HTO pool and atmosphere masses for HTO transport test problem three.
Figure 4. Total HTO pool and atmosphere masses for HTO transport test problem three.
3. OXIDATION CORRELATION UPDATE

The version of MELCOR 1.8.2 used for the ITER-FEAT GSSR\textsuperscript{5} contained oxidation correlations for beryllium, graphite (carbon), and tungsten for steam only and not air. The beryllium correlations in this version of the code did not contain the latest beryllium oxidation rate equations identified in SADL-2000.\textsuperscript{2} The changes documented in this section update these correlations and add those for air.

In the previous version of this code,\textsuperscript{7} FORTRAN coding was added to the MELCOR subroutine HSRUN2 that modified the thickness of a given heat structure at the surface if the material on that surface was either beryllium, graphite, or tungsten and the heat structure surface was exposed to steam. This model assumes that as the material oxidizes it will thin, that is the growth of an oxide film was not accounted for by this model. This thinning was accomplished in the code by changing the heat structure node positions, conduction factors and volumes used by the heat conduction equation solution of the MELCOR code as presented in Reference 8. The FORTRAN coding for these factors was copied from MELCOR subroutine HSGEOM. As the calculation proceeds, this thinning process continues until either all of the surface material on a heat structure or all of the steam in the attached control volume are consumed, and then this FORTRAN coding terminates the oxidation reaction at this heat structure surface. The oxidation correlations for the reactions were contained in a separate subroutine.

In this updated version of MELCOR 1.8.2, the oxidation FORTRAN coding described above was extracted from subroutine HSRUN2 and placed in two separate subroutines, called OXIDIZW for reactions involving steam and OXIDIZA for reactions involving air. These subroutines automatically actuate if steam or oxygen are present in the atmosphere of the control volume associated with a heat structure surface that contains materials designated by the user as either: Beryllium, Carbon, CFC, or tungsten. These new changes have the advantage of restoring subroutine HSRUN2 to near its original state, making the oxidation modifications more transparent to the quality assurance (QA) process when compared against the original version of this subroutine, while requiring the user to only specify the appropriate material name in the heat structure input to actuate this oxidation model. The new subroutines perform the same function as the FORTRAN coding removed from HSRUN2 described above. As before, the reaction rate equations were placed in two additional subroutines, the first containing reactions for beryllium, carbon, CFC and tungsten in water called OXIRATW and the second containing reaction rate equations for beryllium, carbon, CFC and tungsten in air called OXIRATA. A listing of these subroutines and the latest changes made to HSRUN2 appear in Appendix B.

The oxidation rate equations for beryllium in steam contained in OXIRATW as taken from SADL\textsuperscript{3} and converted to kg-Be/m\textsuperscript{2}-s, are:

\[
R_{\text{Be--H}_2\text{O}} = sP_M \left(0.1205 \times e^{-13.465/T}\right) \quad \text{for } T < 773 \text{ K} \quad (3)
\]

\[
R_{\text{Be--H}_2\text{O}} = sP_M \left(4.378 \times 10^{15} \times e^{-42.933/T}\right) \quad \text{for } 773 \text{ K} \leq T \leq 873 \text{ K} \quad (4)
\]

\[
R_1 = 49.398 \times e^{-12.500/T}
\]

\[
R_2 = 2.571 \times 10^7 \times e^{-28.789/T} \quad \text{for } T < 1133 \text{ K}
\]

\[
R_2 = 35.357 \times e^{-13.387/T} \quad \text{for } T \geq 1133 \text{ K}
\]

\[
R_{\text{Be -- H}_2\text{O}} = sP_M \sqrt{R_1 R_2} \quad \text{for } T > 873 \text{ K} \quad (5)
\]
where, s is a safety factor, set equal to two for design basis accidents (DBA) and to one for beyond design basis accidents (BDBA), and \( P_m \) is a scaling factor for steam pressure defined as

\[
P_m = (P_{H_2O}/P_o)^{0.9}
\]  

(6)

where \( P_{H_2O} \) is the transient steam pressure and \( P_o \) is the steam pressure during the reaction tests from which Equations 3-5 were derived (\( P_o = 0.86 \times 10^5 \) Pa).

The oxidation rate equations for carbon or CFC in steam contained in OXIRATW as taken from SADL\(^3\) and converted to \( \text{kg-C/m}^2\text{-s} \), are:

\[
R_{C-H_2O} = sp_m \left(8.571 \times 10^7 \times e^{-70.800/RT}\right) \quad T < 1470 \text{ K}
\]  

(7)

\[
R_{C-H_2O} = sp_m \left(5.196 \times e^{-22.500/RT}\right) \quad T \geq 1470 \text{ K}
\]  

(8)

where, \( R \) is the universal gas constant equal to 1.98 kcal/kg-mole-K, the safety factor, \( s \), is set equal to two for DBA and to one for BDBA, and \( P_m \) is a scaling factor for steam pressure defined as

\[
P_m = (P_{H_2O}/P_o)
\]  

(9)

and where \( P_o = 0.86 \times 10^5 \) Pa.

Finally, the oxidation rate equations for tungsten in steam contained in OXIRATW as taken from SADL\(^3\) and converted to \( \text{kg-W/m}^2\text{-s} \), is:

\[
R_{W-H_2O} = sp_m \left(41.238 \times e^{-33.106/RT}\right) \quad T < 1470 \text{ K}
\]  

(10)

where, the \( s \) is a safety factor is set equal to two for DBA and to one for BDBA, and \( P_m \) is a scaling factor for steam pressure defined as

\[
P_m = (P_{H_2O}/P_o)^{0.78}
\]  

(11)

and where \( P_o = 0.84 \times 10^5 \) Pa.

The oxidation rate equations for beryllium in air contained in OXIRATA as taken from SADL\(^3\) and converted to \( \text{kg-Be/m}^2\text{-s} \), are:

\[
R_{Be-O_2} = sp_m \left(4.833 \times 10^5 \times e^{-26.200/T}\right) \quad T < 1073 \text{ K}
\]  

(12)

\[
R_{Be-O_2} = sp_m \left(34.83 \times e^{-15.900/T}\right) \quad T \geq 1073 \text{ K}
\]  

(13)

where, \( s \) is a safety factor, set equal to five for design DBA and to one for BDBA, and \( P_m \) is a scaling factor for oxygen pressure defined as
\[ P_M = \left( \frac{P_{O2}}{P_o} \right) \]  \hspace{1cm} (14)

where \( P_{O2} \) is the transient oxygen pressure and \( P_o \) is the oxygen pressure during the reaction tests upon which the above equations were derived \( (P_o = 0.181 \times 10^5 \text{ Pa}) \).

The oxidation rate equations for carbon in air contained in OXIRATA as taken from SADL\(^3\) and converted to kg-C/m\(^2\)-s, are:

\[
R_{C-O2} = sP_M \left( 0.2472 \times e^{-5710/T} \right) \quad T < 1273 \text{ K} \quad (15)
\]

\[
R_{C-O2} = sP_M \left( 1.56 \times 10^{-2} \times e^{-2260/T} \right) \quad T \geq 1273 \text{ K} \quad (16)
\]

and for CFC taken from Reference 9 are

\[
R_{C-O2} = sP_M \left( 1.4754 \times 10^{7} \times e^{-26.128/T} \right) \quad T < 983 \text{ K} \quad (17)
\]

\[
R_{C-O2} = sP_M \left( 3.6308 \times 10^{1} \times e^{-13.475/T} \right) \quad 983 \text{ K} \leq T \leq 1448 \text{ K} \quad (18)
\]

\[
R_{C-O2} = sP_M \left( 1.56 \times 10^{-2} \times e^{-2260/T} \right) \quad T > 1448 \text{ K} \quad (19)
\]

where, the safety factor, \( s \), is set equal to two for DBA and to one for BDBA, and \( P_m \) is a scaling factor for oxygen pressure defined as

\[ P_M = \left( \frac{P_{O2}}{P_o} \right) \]  \hspace{1cm} (20)

and where \( P_o = 0.181 \times 10^5 \text{ Pa} \).

Finally, the oxidation rate equations for tungsten in air contained in OXIRATA as taken from SADL\(^3\) and converted to kg-W/m\(^2\)-s, are:

\[
R_{W-O2} = sP_M \left( 1.62 \times 10^{6} \times e^{-24,000/T} \right) \quad T < 973 \text{ K} \quad (21)
\]

\[
R_{W-O2} = sP_M \left( 7.448 \times e^{-12,170/T} \right) \quad T \geq 973 \text{ K} \quad (22)
\]

where, the \( s \) is a safety factor is set equal to two for DBA and to one for BDBA, and \( P_m \) is a scaling factor for oxygen pressure defined as

\[ P_M = \left( \frac{P_{O2}}{P_o} \right)^{0.5} \]  \hspace{1cm} (23)

and where \( P_o = 0.181 \times 10^5 \text{ Pa} \).
4. CORRECTIONS TO THE AEROSOL TRANSPORT SUBROUTINE RN1RN4

Two changes were made to the MELCOR aerosol transport subroutine RN1RN4. The first change modified an aerosol density feature in this subroutine that had previously been specified internally, so as to allow the user to specify this density through user input. The original change was to the user input variable “rhonom” in the MELCOR aerosol transport input. During loss-of-coolant accidents (LOCA) analyses in ITER it was noticed that over a significant period of the transient that the aerosol mass was dominated by water droplets or fog. Often as the accident progressed, the atmosphere would dry out leaving only the aerosol particulate of interest, which is tungsten in the case of ITER. Because aerosol density greatly affects the settling rate of aerosols, and because MELCOR only allowed the user to specify a single aerosol density (rhonom), the subroutine RN1RN4 was changed internally to give a homogenous density based on water droplet and tungsten aerosol masses. Unfortunately, this change did not allow the user to specify an aerosol density other than for tungsten.

The first modification uses the input value of rhonom to calculate the homogenous or nominal aerosol density as follows:

\[
\rho_{\text{nom}} = x_{\text{H}_2\text{O droplets}} \times \rho_{\text{H}_2\text{O}} + \left(1 - x_{\text{H}_2\text{O droplets}}\right) \rho_{\text{rhonom}}^{\text{user input}}
\]

where the mass fraction of water droplets (\(x_{\text{H}_2\text{O}}\)) in the atmosphere of a given control volume is defined as:

\[
x_{\text{H}_2\text{O droplets}} = \frac{\sum_{i=1}^{\text{nbins}} m_{14,i}}{\sum_{i=1}^{\text{nclass}} \sum_{i=1}^{\text{nbins}} m_{1,i}}
\]

The nomenclature “nclass” and “nbins” represents the number of aerosol material classes (class 14 is reserved by MELCOR for water droplets) and number of aerosol mass bins specified by the user for a given problem.

The second modification introduced a logical variable into RN1RN4 called “STDMEL” that switches the coding in this subroutine, upon compilation, back to its original function. That is, switches back to the assumption that only air, at standard pressure, exists in the atmosphere of all control volumes for calculating the aerosol transport coefficients during accident analyses. This change was made to allow for comparisons against the original version of MELCOR 1.8.2 code. The FORTRAN listing of these changes appears in Appendix C.
5. CORRECTIONS TO THE AIR CONDENSATION SUBROUTINES

One of the modifications made to the MELCOR 1.8.2 code for application to fusion reactors was the addition of an air condensation model.\textsuperscript{11} The reason for adding this model was to assess the impact of air entering the cryostat during accident analyses. Because the ITER magnets are at cryogenic temperatures, any air entering the cryostat would condense and freeze onto the surface of these magnets. The result would be to slow the pressurization event that ensues during air ingress accidents in the cryostat. Upon reexamination of this model, during the process of pedigreeing a version of the MELCOR 1.8.2 for ITER, it was discovered that the polynomial equations used by this model to calculate the nitrogen and oxygen saturated liquid enthalpy had different reference energies than the non-condensable equations used by the MELCOR 1.8.2 code to calculate the nitrogen and oxygen vapor enthalpy. The result was an incorrect estimate of the energy change by this model during condensation of these two gases. The affected equation of this model, restated here, is as follows:

\[ q = \Gamma_c \cdot \left( h_v - h_{1\text{sat}} \right)_{O_2} + \Gamma_c \cdot \left( h_v - h_{1\text{sat}} \right)_{N_2} \]  

(26)

where,

- \( \Gamma_c \) = condensation mass flux (kg/m\(^2\)-s)
- \( h_v \) = vapor enthalpy (J/kg)
- \( h_l \) = liquid enthalpy (J/kg)
- \( O_2, N_2 \) = represent oxygen and nitrogen
- \( \text{sat} \) = saturated state

To remedy this inconsistency, the following equations were adopted in place of Equation 26:

\[ q = \Gamma_c \cdot \left( h_{v,1\text{sat}} \right)_{O_2} + \Gamma_c \cdot \left( h_{v,1\text{sat}} \right)_{N_2} \]  

(27)

with

\[ h_{v,1} = c_p \cdot \left( T_v - T_1 \right) + h_{fg,\text{sat}} \]  

(28)

where the vapor specific heat capacity, \( c_p \) (J/kg-K), for oxygen and nitrogen are the ideal values for these gases as presented in Reference 12, which are 916.9 J/kg-K and 1038.3 J/kg-K, respectively, and the heat of vaporization, \( h_{fg} \) (J/kg) is calculated from Clapeyron’s Equation as follows:

\[ h_{fg,\text{sat}} = T \left( \frac{1}{\rho_v} - \frac{1}{\rho_f} \right) \frac{dp}{dT_{\text{sat}}} \]  

(29)

where

- \( T \) = saturation temperature/heat structure surface temperature (K)
- \( \rho_v \) = vapor density (kg/m\(^3\))
- \( \rho_f \) = saturated liquid density (kg/m\(^3\))
- \( \frac{dp}{dT_{\text{sat}}} \) = derivative of the pressure with respect to temperature at saturation (Pa/K)

The derivative of saturation pressure with respect to temperature was found by numerically differentiating the saturation pressure polynomials presented in Reference 13 for oxygen or nitrogen. The gas density is defined from the ideal gas law as follows:

\[ \rho_v = \frac{RT}{p} \]  

(29)
where $R$ is the Gas Constant (J/kg-K) for that gas, $T$ is the gas temperature (K), and $p$ is the gas pressure (Pa). The remaining parameter that needs to be defined is the saturated liquid density which was evaluated from the polynomials presented in Reference 13, which take the following form:

$$
\rho_f = \sum_{i=1}^{6} D_i \left(1 - \frac{T}{T_c}\right)^{(i-1)/3}
$$

(30)

where the polynomial coefficients, $D_i$, and critical temperatures, $T_c$ (K), are also given in Reference 13.

To verify that the energy change predicted by this model approximates that predicted by a more complete Equation of State (EOS), property values were selected at a pressure of $5.0 \times 10^5$ Pa and temperatures that appear in Reference 13 and compared to that predicted made by this model. The results appear in Tables 1 and 2. As can be seen, the comparison is very good.

Most of the modifications for the air condensation model have been made to MELCOR subroutines HSRUN2, HSTRAN, and HSENST. These modifications are listed in Appendix D, along with the condensation subroutines that have been added to the code. The changes to subroutines HSRUN2, HSTRAN, and HSENST are different from those described in Reference 11, and were implemented in order to return subroutine HSRUN2 closer to its’ original state for ease in QA checking and new condensation subroutines have been developed to correct the energy error described above. A final change for this model relative to that described in Reference 10 has to do with the calculation of the binary gaseous diffusion coefficients. Reference 10 developed a subroutine call DABMIX. It was later discovered that the MELCOR code performed the same operations as DABMIX, so the standard MELCOR treatment was adopted by removing subroutine DABMIX and replacing the diffusion coefficient correlation in MELCOR subroutine MPVIS with that of Wilke and Lee (note Equation 21 of Reference 10).
Table 1. Comparison of condensation model enthalpy change with values from Reference 10 for oxygen

<table>
<thead>
<tr>
<th>$T_{\text{liq}}$ (K)</th>
<th>$h_{l,v}$ (kJ/kg) 200</th>
<th>$h_{l,v}$ (kJ/kg) 300</th>
<th>$h_{l,v}$ (kJ/kg) 400</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td>Ref. 7</td>
<td>Model</td>
</tr>
<tr>
<td>60</td>
<td>238.26</td>
<td>238.26</td>
<td>366.63</td>
</tr>
<tr>
<td>70</td>
<td>230.50</td>
<td>230.50</td>
<td>349.70</td>
</tr>
<tr>
<td>80</td>
<td>222.30</td>
<td>222.30</td>
<td>332.33</td>
</tr>
<tr>
<td>90</td>
<td>213.23</td>
<td>213.23</td>
<td>314.09</td>
</tr>
</tbody>
</table>

Table 2. Comparison of condensation model enthalpy change with values from Reference 10 for nitrogen

<table>
<thead>
<tr>
<th>$T_{\text{liq}}$ (K)</th>
<th>$h_{l,v}$ (kJ/kg) 200</th>
<th>$h_{l,v}$ (kJ/kg) 300</th>
<th>$h_{l,v}$ (kJ/kg) 400</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td>Ref. 7</td>
<td>Model</td>
</tr>
<tr>
<td>63.15</td>
<td>214.83</td>
<td>214.83</td>
<td>360.19</td>
</tr>
<tr>
<td>70</td>
<td>203.14</td>
<td>208.14</td>
<td>338.12</td>
</tr>
<tr>
<td>80</td>
<td>195.65</td>
<td>195.65</td>
<td>320.25</td>
</tr>
<tr>
<td>90</td>
<td>180.64</td>
<td>180.64</td>
<td>294.85</td>
</tr>
</tbody>
</table>
6. REFERENCES


Appendix A

Code Listing of Changes to MELCOR for HTO Transport Model
******** added after line 458

DIMENSION FTHTX(999), FTTHM(999), FTEXC(999)

******** added after line 1139

DO 35 NVOL = 1, NUMVOL
   FTHTX(NVOL) = ZN(1,NVOL,13)-ZN(1,NVOL,7)
   .            + ZN(2,NVOL,13)-ZN(2,NVOL,7)
   IF(FTHTX(NVOL).LE.0.0) THEN
      IF(ZN(1,NVOL,7)+ZN(2,NVOL,7).NE.0.0) THEN
         FTHTX(NVOL) = FTHTX(NVOL)/(ZN(1,NVOL,7)+ZN(2,NVOL,7))
      ELSE
         FTHTX(NVOL) = 0.0
      ENDIF
   ELSE
      IF(ZN(3,NVOL,7).NE.0.0) THEN
         FTHTX(NVOL) = FTHTX(NVOL)/ZN(3,NVOL,7)
      ELSE
         FTHTX(NVOL) = 0.0
      ENDIF
   ENDIF
35 CONTINUE

*******added after line 1205

DO 45 NVOL = 1, NUMVOL
   FTTHM(NVOL) = (XMASSN(1,NVOL)-ZN(1,NVOL,13)
   .         + (XMASSN(2,NVOL)-ZN(2,NVOL,13)))
   IF(FTTHM(NVOL).LE.0.0) THEN
      IF(ZN(1,NVOL,13)+ZN(2,NVOL,13).NE.0.0) THEN
         FTTHM(NVOL) = FTTHM(NVOL)/(ZN(1,NVOL,13)+ZN(2,NVOL,13))
      ELSE
         FTTHM(NVOL) = 0.0
      ENDIF
   ELSE
      IF(ZN(3,NVOL,13).NE.0.0) THEN
         FTTHM(NVOL) = FTTHM(NVOL)/ZN(3,NVOL,13)
      ELSE
         FTTHM(NVOL) = 0.0
      ENDIF
   ENDIF
45 CONTINUE

*******added after line 1423

DO 55 NVOL = 1, NUMVOL
   FTTHM(NVOL) = XMASSN(1,NVOL)-ZN(1,NVOL,9)
   .         + (XMASSN(2,NVOL)-ZN(2,NVOL,9))
   IF(FTTHM(NVOL).LE.0.0) THEN
      IF(ZN(1,NVOL,9)+ZN(2,NVOL,9).NE.0.0) THEN
         FTTHM(NVOL) = FTTHM(NVOL)/(ZN(1,NVOL,9)+ZN(2,NVOL,9))
      ELSE
         FTTHM(NVOL) = 0.0
      ENDIF
   ELSE
      IF(ZN(3,NVOL,13).NE.0.0) THEN
         FTTHM(NVOL) = FTTHM(NVOL)/ZN(3,NVOL,9)
      ELSE
         FTTHM(NVOL) = 0.0
      ENDIF
   ENDIF
55 CONTINUE

******* added after line 1728

154 DO 155 NVOL=1,NUMVOL
DO 155 NMAT=1,3
    ZN(NMAT,NVOL,?) = XMASSN(NMAT,NVOL)
155 CONTINUE

****** added after line 1756
C
DO 165,NVOL=1,NUMVOL
    FTEXC(NVOL) = XMASSN(1,NVOL)-ZN(1,NVOL,7) + XMASSN(2,NVOL)-ZN(2,NVOL,7)
    IF(FTEXC(NVOL).LE.0.0) THEN
        IF(ZN(1,NVOL,7)+ZN(2,NVOL,7).NE.0.0) THEN
            FTEXC(NVOL) = FTEXC(NVOL)/(ZN(1,NVOL,7)+ZN(2,NVOL,7))
        ELSE
            FTEXC(NVOL) = 0.0
        ENDIF
    ELSE
        IF(ZN(3,NVOL,7).NE.0.0) THEN
            FTEXC(NVOL) = FTEXC(NVOL)/ZN(3,NVOL,7)
        ELSE
            FTEXC(NVOL) = 0.0
        ENDIF
    ENDIF
165 CONTINUE

****** added after line 1811
C
    FTTHM(NVOL) = FTTHM(NVOL) + FTEXC(NVOL)

****** added after line 1812
C
C + + PRE-CONVECTION TRANSFER
    IF (MONRN2.NE.0) THEN
        CALL RN2BOIL(FTTHM,DTSUB,NUMVOL)
    ENDIF

****** added after line 1825
C
C + + POST-CONVECTION TRANSFER
    IF (MONRN2.NE.0) THEN
        CALL RN2BOIL(FTTHM,DTSUB,NUMVOL)
    ENDIF

END OF CHANGES TO CVHRN3
***** added after line 125

C -*- INCLUDE DCHDB
C
C COMMON WITH POINTERS TO BEGINNING OF DECAY HEAT DATA BASE
C
COMMON /DCHDB/ IRDCHF, IRDCHN, IIDCHF, IIDCHN,
   1 ILDCHF, ILDCHN, ICDCHF, ICDCHN
C
C IRDCHF = THE FIRST ELEMENT OF THE REAL ARRAY CONTAINING
C          THE DECAY HEAT DATA.
C IRDCHN = THE NUMBER OF ELEMENTS OF THE REAL ARRAY
C          CONTAINING DECAY HEAT DATA.
C
C IIDCHF = THE FIRST ELEMENT OF THE INTEGER ARRAY CONTAINING
C          THE DECAY HEAT DATA.
C IIDCHN = THE NUMBER OF ELEMENTS OF THE INTEGER ARRAY
C          CONTAINING DECAY HEAT DATA.
C
C ILDCHF = THE FIRST ELEMENT OF THE LOGICAL ARRAY CONTAINING
C          THE DECAY HEAT DATA.
C ILDCHN = THE NUMBER OF ELEMENTS OF THE LOGICAL ARRAY
C          CONTAINING DECAY HEAT DATA.
C
C ICDCHF = THE FIRST ELEMENT OF THE CHARACTER (*8) ARRAY
C          CONTAINING THE DECAY HEAT DATA.
C ICDCHN = THE NUMBER OF ELEMENTS OF THE CHARACTER (*8) ARRAY
C          CONTAINING DECAY HEAT DATA.
C
********** change to line 142

+ MAXSCR, XREALX(IRELCS), MAXSCI, INTEGE(IINTCS),
+ ICDCHN, CHARAC(ICDCHF), IIDCHN, INTEGE(IIDCHF) )

END OF CHANGES TO RNIDBC
************** changes to line 13

+ IIHS, IHS, IRSC, XSC, IISC, ISC, NDCHC, CDCH, IIDCH, IDCH)

C

C * * HTO MODS TO GET AT CLASS NAMES (BJM 2/20/07)
CHARACTER * 8 CDCH(NDCHC)
DIMENSION IDCH(IIDCH)

C

* INCLUDE DCHPNT

C

COMMON BLOCK WITH DECAY HEAT DATABASE POINTERS

C SET UP BY JOHN ORMAN

C VERSION 1.06 12/1/86

C

COMMON /DCHPNT/

**** CLASS POINTERS ****

1 MFPCNM, MFFCID, MCLSMS, MCLSEL, MCLEPT, MCLELM,
2 MDCHET, MDCTIM, MTIMDH, MTMDHL, MTDALL,
3 MCLSPB, MCLSPE, MCLSPO, MCLSPN,
4 NDFPCL, NDELEM, NDCTIM, NTDALL,
5 NFPCLS, NELEMS, NDCTIM, NTDALL,
6 IOSDCH

C

**** CLASS POINTER DESCRIPTIONS *****

C

*** POINTERS TO DATA ARRAYS ***

C

MFPNAM - FISSION PRODUCT CLASS NAMES - VARIABLE NAME = FPCNAM

C

************** added after line 703

+ NDFPCL, NFPCLS, CDCH(MFPCNM), IDCH(MFPCID) 

END OF CHANGES TO RN1DBD
****** added after line 18
+ NDFPCL, NFPCLS, FPCNAM, IDCH )
C
C   * HTO MODS TO GET AT CLASS NAMES (BJM 2/20/07)
CHARACTER * 32 FPCNAM(NDFPCL)
DIMENSION IDCH(NFPCLS)

****** added after line 291
C  ---- HTO TRANSPORT FLAG, STANDARD AIR & GRAVITY ONLY FLAGS
C
LOGICAL GVONLY, FLGHTO, STDMEL
DATA GVONLY /.TRUE./
DATA FLGHTO /.FALSE./
DATA STDMEL /.FALSE./

****** added after line 313
C
FLAGGV = 0.
IF(.NOT.STDMEL) THEN
IF(GVONLY) FLAGGV = 1.
C
C   * CHECK FOR HTO (MUST BE LAST CLASS IN)
DO J=1,NCLS
IF(FPCNAM(J)(1:3).EQ.'HTO') THEN
IF(IDCH(J).NE.NCLS) THEN
PRINT *,"HTO MUST BE LAST CLASS FOR MODEL TO WORK"
STOP
ENDIF
FLGHTO = .TRUE.
ENDIF
ENDDO

****** added after line 368
C
C   IF HTO TRANSPORT SKIP DEPOSITION (BJM 2/20/07)
NCLSB = NCLS
IF(FLGHTO) NCLSB = NCLS - 1

****** added after line 413
C
C   INCLUDE HTO AEROSOL SOURCES
C
IF(FLGHTO) THEN
C
DO 141 I = 1, NSRCA
IF(ISRCA(3,I) .NE. NCLS) GO TO 141
IF(ISRCA(1,I) .NE. N) GO TO 141
J1 = ISRCA(3,I)
IF(ISRCA(2,I) .EQ. 1) THEN
C
IF(CVOL(1,N).GT.RNDNUM) THEN
AER1LN( J1,N) = MAX(0.,AER1LN( J1,N)+SRCA( 2,I)*DT)
ELSE
AER1GN(1,J1,N) = MAX(0.,AER1GN(1,J1,N)+SRCA( 2,I)*DT)
ENDIF
C
ELSE
C
IF(CVOL(3,N).GT.RNDNUM) THEN
DO 142 J = 1, NSEC
AER1GN(J,J1,N) = MAX(0.,AER1GN(J,J1,N)+SRCA(J+1,I)*DT)
ALPHA=0.D0
DO 143 J = 1, NSEC
ALPHA = ALPHA + SRCA(J+1,I)*DT
143 CONTINUE
AER1LN( J1,N) = MAX(0.,AER1LN( J1,N)+ALPHA)
ENDIF

C

141 CONTINUE
C
C ----- IF HTO TRANSPORT, INCLUDE CONDENSATION SKIP DEPOSITION
C ----- CONDENSATION ON HS DIRECT TO LIQUID PHASE
C
C IF(CVOL(1,N).GT.RNDNUM) THEN
HTOCON = 0.
DO 396 I = ISUM, ISUMP
IF(WCOND(I).GE.ZERO .AND. XMASS(3,N).GT.0.0) THEN
   HTOCON = HTOCON + DT*WCOND(I)*AREAHS(I)/XMASS(3,N)
END IF
396 CONTINUE
IF(HTOCON.GT.0.0) THEN
   K = NCLS
   DAERLN = 0.
   DO 398 J = 1, NSEC
      DAERGN = HTOCON*AER1GN(J,K,N)
      DAERGN = MIN(AER1GN(J,K,N),DAERGN)
      DAERLN = DAERLN + DAERGN
   AER1GN(J,K,N) = MAX(0.,AER1GN(J,K,N)-DAERGN)
398 CONTINUE
   AER1LN( K,N) = AER1LN( K,N)+DAERLN
397 CONTINUE
ENDIF
ENDIF
C

**** change to line 465
DO 150 J = 1, NCLSB

***** change to line 500
C
DO 160 J = 1, NCLSB

****** change to line 518
C
DO 165 J = 1, NCLSB

****** change to line 714
C
DO 470 J = 1, NCLSB

****** change to line 805
C
DO 1581 J=1,NCLSB

END OF CHANGES TO RN1RN4
ADDITIONAL SUBROUTINES

*DECK RN2BOIL
CMP$ /RN2MOVMK/ 1/05/93 08:36:04 SLT
SUBROUTINE RN2BOIL(FCBOIL,DTSUB,NNVOL)
C
C       RUN ROUTINE FOR THE RN2 PACKAGE FOR MELCOR
C
C       INPUT - NONE
C
C       OUTPUT - AVAILABLE SCRATCH SPACE
C
C
SAVE
C
C*** DIMENSION THE ARRAYS IN THE CALLING ARGUMENT (WHICH IS CALLED
C*** FROM CVHRN3)
C
C DIMENSION FCBOIL(*)
C
*- INCLUDE BLANK
C
C     MAIN DATABASE COMMON BLOCKS
C     REL = REALS, INT = INTEGERS, CHR = CHARACTERS, LOG = LOGICALS
C     NUM*** = DIMENSION OF ARRAY
C     NED*** = PHYSICS PACKAGES WORKING + SCRATCH STORAGE
C     IXXXCS = POINTER TO START OF SCRATCH STORAGE
C     I***ES = POINTER TO START OF EDIT PACKAGE SCRATCH STORAGE
C     XREALX, INTEGE, LOGICA, CHARAC STORAGE ARRAYS
C
C     NUMREL = MAXIMUM NUMBER OF REALS            (DIMENSION OF XREALX)
PARAMETER (NUMREL = 500000)
C     NUMINT = MAXIMUM NUMBER OF INTEGERS         (DIMENSION OF INTEGE)
PARAMETER (NUMINT = 50000)
C     NUMLOG = MAXIMUM NUMBER OF LOGICALS         (DIMENSION OF LOGICA)
PARAMETER (NUMLOG = 5000)
C     NUMCHR = MAXIMUM NUMBER OF CHARACTER*8 VARIABLES  (DIM OF CHARAC)
PARAMETER (NUMCHR = 30000)
C
C     DECLARING DREALX IN COMMON ENSURES THAT STORAGE BEGINS AT THE
C     CORRECT 8-BYTE BOUNDARY ON THOSE MACHINES THAT CARE
C
COMMON /DBREAL/ NEDREL, IRELCS, IRELES
COMMON / /                          DREALX(NUMREL/2)
DOUBLE PRECISION DREALX
COMMON /DBINTG/ NEDINT, IINTCS, IINTESS, INTEGE(NUMINT)
COMMON /DLOGCC/ NEDLOG, ILOGCS, ILOGES, LOGICA(NUMLOG)
LOGICAL LOGICA
COMMON /DBCHAD/ NEDCHR, ICHRCS, ICHRRES
COMMON /DBCHAR/                          CHARAC(NUMCHR)
CHARACTER*8 CHARAC
C
C     SINGLE PRECISION REALS
C
DIMENSION XREALX(NUMREL)
EQUIVALENCE (XREALX(1), DREALX(1))
C
C     VARIABLE PRECISION REALS - DOUBLE FOR 32 BIT MACHINES
C     SINGLE FOR 64 BIT MACHINES
C
C     DOUBLE PRECISION VREALX(NUMREL/2)
EQUIVALENCE (VREALX(1), DREALX(1))

*-

*- INCLUDE RN2DB
C
C     RN2 COMMON BLOCK FOR NUMBER OF VARIOUS DATA TYPES
C     DIMENSIONS OF CALCULATIONAL SCRATCH ADDED 14-NOV-90
C
COMMON/RN2DB/ IRRN2F, IRRN2N, IIRN2F, IIRN2N,
+     ILRN2F, ILRN2N, ICRN2F, ICRN2N,
+     NRRN2S, NIRN2S

*-
* INCLUDE RN1DB
C
C RN1 COMMON BLOCK FOR NUMBER OF VARIOUS DATA TYPES
C
COMMON/RN1DB/ IRRN1F, IRRN1N, IIRN1F, IIRN1N,
+ ILRN1F, ILRN1N, ICRN1F, ICRN1N
*-

* INCLUDE CVHDB
C
C POINTERS TO CONTROL VOLUME HYDRODYNAMICS DATA BLOCKS IN
C DATABASE COMMONS
C DIMENSIONS OF CALCULATIONAL SCRATCH ADDED 14-NOV-90
C
COMMON /CVHDB/ IRCVHF, IRCVHN, IICVHF, IICVHN,
1 ILCVHF, ILCVHN, ICCVHF, ICCVHN,
2 NRCVHS, NICVHS
C

* INCLUDE FLDB
C
C POINTERS TO FLOW PATH DATA BLOCKS IN DATABASE COMMONS
C DIMENSIONS OF CALCULATIONAL SCRATCH ADDED 14-NOV-90
C
COMMON /FLDB/ IRFLF, IRFLN, IIFLF, IIFLN,
1 ILFLF, ILFLN, ICFLF, ICFLN,
2 NRFLS, NIFLS
C

* INCLUDE RN2CPU
COMMON/RN2CPU/CPUR2C,CPUR2E,CPUR2R
*-
C
* INCLUDE DCHDB
C
C COMMON WITH POINTERS TO BEGINNING OF DECAY HEAT DATA BASE
C
COMMON /DCHDB/ IRDCHF, IRDCHN, IIDCHF, IIDCHN,
1 ILDCHF, ILDCHN, ICDCHF, ICDCHN
C
IRDCHF = THE FIRST ELEMENT OF THE REAL ARRAY CONTAINING
THE DECAY HEAT DATA.
C
IRDCHN = THE NUMBER OF ELEMENTS OF THE REAL ARRAY
CONTAINING DECAY HEAT DATA.
C
IIDCHF = THE FIRST ELEMENT OF THE INTEGER ARRAY CONTAINING
THE DECAY HEAT DATA.
C
IIDCHN = THE NUMBER OF ELEMENTS OF THE INTEGER ARRAY
CONTAINING DECAY HEAT DATA.
C
ILDCHF = THE FIRST ELEMENT OF THE LOGICAL ARRAY CONTAINING
THE DECAY HEAT DATA.
C
ILDCHN = THE NUMBER OF ELEMENTS OF THE LOGICAL ARRAY
CONTAINING DECAY HEAT DATA.
C
ICDCHF = THE FIRST ELEMENT OF THE CHARACTER (*8) ARRAY
CONTAINING THE DECAY HEAT DATA.
C
ICDCHN = THE NUMBER OF ELEMENTS OF THE CHARACTER (*8) ARRAY
CONTAINING DECAY HEAT DATA.
C

CPU TIME AT END
C
CALL MXXCPU(CFUBEG)
C
CALL RN2BOL1(FCBOIL, DTSUB, NNVOL,
+ IRRN1N, XREALX(IRRN1F), IIRN1N, INTEGE(IIRN1F),
+ ICDCHN, CHARAC(ICDCHF), IIDCHN, INTEGE(IIDCHF) )
C
CPU TIME AT END
CALL MXXCPU(CPUEND)
CPUR2C=CPUR2C+CPUEND-CPUBEG

RETURN
END

*DECK RN2BOL1

SUBROUTINE RN2BOL1(FCBOIL, DTSUB, NNVOL,
+      NRRN1, XRN1, NIRN1, IRN1, NDCHC, CDCH, IIDCH, IDCH)

CHARACTER * 8 CDCH(NDCHC)
DIMENSION IIDCH(IIDCH)

* - INCLUDE DCHPNT

COMMON BLOCK WITH DECAY HEAT DATABASE POINTERS
SET UP BY JOHN ORMAN
VERSION 1.06 12/1/86

COMMON /DCHPNT/

**** CLASS POINTERS ****
1 MFPCNM, MFFCID, MCLSMS, MCLSEL, MCLEPT, MCLELM,
2 MDCHET, MDCTIM, MTIMDH, MTMDHL, MTDALL,
3 MCLSPB, MCLSPE, MCLSPO, MCLSPN,
4 NDFPCL, NDELEM, NDCTIM, NTDALL,
5 NFPCLS, NELEMS, NDCTIM, NTDALL,
6 IGSDC

**** CLASS POINTER DESCRIPTIONS ******

*** POINTERS TO DATA ARRAYS ***

MFPCNM - FISSION PRODUCT CLASS NAMES - VARIABLE NAME = FPCNAM

SECOND LEVEL DATA BASE MANAGER ROUTINE
CALLED BY RN2MOV SUBROUTINE

INPUT
NRRN2 - NUMBER OF REALS IN RN2 DATABASE
XRN2 - REAL ARRAY OF RN2 DATABASE VARIABLES
NRRN1 - NUMBER OF REALS IN RN1 DATABASE
XRN1 - REAL ARRAY OF RN1 DATABASE VARIABLES
NIRN1 - NUMBER OF INTEGERS IN RN1 DATABASE
IRN1 - INTEGER ARRAY OF RN1 DATABASE VARIABLES
NISC - NUMBER OF INTEGERS IN SCRATCH DATABASE
ISC - INTEGER ARRAY OF SCRATCH DATABASE

OUTPUT
NONE

EXTERNAL CALLS - RN2MOX

* - INCLUDE SRM1B

MELCOR/SR MOD-1
MODIFICATION FOR RN FILTER MODEL
MELCOR REFERENCE VERSION: MELCOR V1.80 (SNL)
MELCOR/SR CYCLE: MELCOR/SR MOD-1 CYCLE B
UPDATE SEQUENCE IDENTIFIER: SRMOD1B

* -

DIMENSION XRN1(NRRN1),IRN1(NIRN1),FCBOIL(*)

SAVE
*- INCLUDE RN2PNT
C   RN2 DATABASE POINTERS
C
COMMON/RN2PNT/ IT2ARG, IT2ARL, IT2AFG, IT2AFL,
+        IT2VPG, IT2VPL, IT2VFG, IT2VFL,
+        IR2ARG, IR2ARL, IR2AFG, IR2AFL,
+        IR2VPG, IR2VPL, IR2VFG, IR2VFL,
+        IORRN2, IOIRN2
-*

*: INCLUDE RN2PLS
C   RN2 DATABASE FOR POOL SCRUBBING AEROSOL REMOVAL
C
COMMON/RN2PLS/IDBIN, IRRAT, IVRISE, IVSRWM, INUMFP, NUMPLS, NPLS
*

`: INCLUDE RN2FLT
C   RN2 DATABASE FOR FILTER AEROSOL AND FP VAPOR REMOVAL
C
COMMON/RN2FLT/INMFLT, IDF, IFLTTP, IDFFTPO, IDFFTIV,
+        IMFPCA, IMFPTA, IMFPTCV, IMFPTTV,
+        IAEFTO, IAEFTN, IRAFTO, IRAFTN, IVPFTO, IVPFTN, IRVFTO, IRVFTN, IOFLT,
+        NUMFLT, NFLT, ISRFLO, ISRFLN, IOSRFL
*

`: INCLUDE RN1PNT
C   RN1 DATABASE POINTERS
C
COMMON/RN1PNT/ IPMSEC,
+        IT1AGO, IT1ALO, IT1VGO, IT1VLO, ITADPO, ITVDPO,
+        IT1AGN, IT1ALN, IT1VGN, IT1VLN, ITADPN, ITVDPN,
+        IR1AGO, IR1ALO, IR1VGO, IR1VLO, IRADPO, IRVDPO,
+        IR1AGN, IR1ALN, IR1VGN, IR1VLN, IRADPN, IRVDPN,
+        IPMCR, IPMCGO, IPMCGN, IPMCVN, IPMADN, IPMADV,
+        IPCLCO, IPMSRA, IPMSSV, IPMSRA, IPMSSV,
+        IPMSRA, IPMRCA, IPMRRSP, IPMRRSP,
+        IPMCN2B, IPMNC2A, IPMNC2B, IPMNCN, IPMNC4, IPMCEN,
+        IPMCRW, IPMVDS, IPMDSV, IPFRCR, IPFRMO, IPFRSMN,
+        IPRCUM, IPRCAV, IPRFDDI, IPRMRLO, IPRMLN, IPRRCCL,
+        IPMCNR, IPMVCN, IPMVNUM, IPMVNN, IPMFDO, IPMFNDN,
+        IPMSC, IPMCN2L, IPMCN2B, IPMCN2B, IPMCN2N, IPMCN2N,
+        NUMSECN, NUMCECN, NUMCLC, NUMCSA, NUMCSV,
+        NNSRC, NNSRV, NUMCOF, NUMSURA, NNSURA, NFLAGS,
+        NCNODE, NCMATA, NCLC, NCLCT, NNCLC, NNCLCT,
+        NVANCL, IPMCRI, ICVPTH,
C PARAMETERS ADDED BY R.K.COLE, JULY 1991
C POINTERS ADDED BY R.C.SMITH, ST. PATTY'S DAY 1992
+        INAXLC, INRADC, NGAPC, KCMPC, NFMATC, LXMFCV,
C
*: INCLUDE MXXTIM
C   TIME COMMON BLOCK
C   TIME = PROBLEM TIME
C   DT = TIMESTEP
C   CPU = CPU TIME
C   NCYCLE = CYCLE NUMBER
C   NDTPAK = PACKAGE CONTROLLING TIMESTEP
COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
COMMON /MXXTPK/ NDTPAK
CHARACTER*3 NDTPAK
*

*: GET OFFSETS, ETC
C
IF(MOD(NCYCLE,2).EQ.0) THEN
C   EVEN CYCLE
C   J2=0

26
IF=0
J1=0

ELSE

C       ODD CYCLE

C
J2=IORRN2
IF=IOFLT
J1=IORRN1

END IF

C       CALL RN2MOX

C       FLOW PATH PROCESSES – CONVECTION ONLY

CALL RN2BOL2(FCBOIL, NNVOL,NUMSEC,NUMCLS,DTSUB, +  XRN1(IT1AGN-J1),XRNX1 (IT1ALN-J1), +  NDFPCL, NFPCLS, CDCH(MFPCNM), IDCH(MFPCID))

RETURN

END

*DECK RN2BOL2
SUBROUTINE RN2BOL2(FCBOIL, NNVOL, NSEC, NCLS, DTSUB, +  AER1G, AER1L, +  NDFPCL, NFPCLS, FPCNAM, IDCH )

C

C * * HTO MODS TO GET AT CLASS NAMES (BJM 2/20/07)
CHARACTER * 32 FPCNAM(NDFPCL)
DIMENSION IDCH(NFPCLS)

CSR
INCLUDED MXXTIM SO THAT TIME AND DT CAN BE USED IN RN2DCF CALL
*- INCLUDE MXXTIM
C TIME COMMON BLOCK
C TIME = PROBLEM TIME
C DT = TIMESTEP
C CPU = CPU TIME
C NCYCLE = CYCLE NUMBER
C NDTPAK = PACKAGE CONTROLLING TIMESTEP
COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
COMMON /MXXTPK/ NDTPAK
CHARACTER*3 NDTPAK

*-
*- INCLUDE ICVFLG
C THIS PARAMETER STATEMENT CONTAINS POINTERS TO THE CONTROL
C VOLUMES FLAGS AND SWITCHS IN ARRAY ICVFLG(KCFLDM,NNVOL)
C
C KCVFF POINT TO THE FLOW GEOMETRY FLAG
C KFPSW POINTS TO THE POOL/NO POOL, FOG/NO FOG SWITCH
C KCVTHR POINTS TO THE CONTROL VOLUME THERMO SWITCH
C KCVTYP POINTS TO THE CONTROL VOLUME TYPE
C KCVACT POINTS TO THE ACTIVE/INACTIVE SWITCH
C KCFLDM IS THE FIRST DIMENSION OF THE ICVFLG ARRAY
PARAMETER ( KCVFF = 1, KFPSW = 2, KCVTHR = 3, + 1 KCVTP = 4, KCVACT = 5, KCFLDM = 5 )

*-
*- INCLUDE FLGEO
C THIS PARAMETER STATEMENT CONTAINS POINTERS USED WITH THE
C FLGEO ARRAYS.
C KFHGTF = FLOW PATH HEIGHT IN THE FORWARD DIRECTION
C KFHGTR = FLOW PATH HEIGHT IN THE REVERSE DIRECTION
C KZFM = ELEVATION OF JUNCTION TO THE 'FROM' CONTROL VOLUME
C KZTO = ELEVATION OF JUNCTION TO THE 'TO' CONTROL VOLUME
C KZBJF, KZTJF, KZBJT, KZTJT, KL2PF ADDED 14-APR-92 BY R.K.COLE
**ADVECTION PORTION OF RUN ROUTINE FOR RN2 PACKAGE**

**CODED BY ROBERT J. GROSS, OCTOBER 1990**

**BASED ON ORIGINAL RN2RN1 CODED BY STEPHEN W. WEBB, JUNE 1986**

**CALLED THROUGH RN2MOV AND RNMOW SEQUENCE**

**INPUT**

- **NNVOL** - Number of control volumes
- **NSEC** - Number of sections
- **NCLS** - Number of classes
- **AER1G** - Aerosol masses in gas phase from RN1 package, KG, (NSEC,NCLS,NNVOL) values
- **AER1L** - Aerosol masses in liq phase from RN1 package, KG, (NCLS,NNVOL) values

**OUTPUT**

- **AER1G** - Aerosol masses in gas phase accounting for flow path convective transport, KG, (NSEC,NCLS,NNVOL) values
- **AER1L** - Aerosol masses in liq phase accounting for flow path convective transport, KG, (NCLS,NNVOL) values (NUMFL) values

**SAVE**

*- INCLUDE MXXDTI

**MXXDTI** contains timestep information

- **DTI** (DTOLDI) is the timestep for the present (last) cycle
- **DTMAXI** (DTMINI) is the maximum (minimum) allowed timestep

*- INCLUDE INOUT

**COMMON** /INOUT/ NDIAG,NOUT,NPLOTA,NPLOTB,NREST,NTERMO,NTERMI, NMESAG

1 NMESAG

*- INCLUDE CONST

**C*************** PROCESSOR DEPENDENT NUMBERS**

- **BIGNMP** (BIGNMM) = largest positive (negative) number
- **SMLNMP** (SMLNMM) = smallest positive (negative) number
- **RNDNUM** is the roundoff associated with the number 1.

**C*************** CONSTANTS**

- **GRAVITY** is the acceleration of gravity (M/S**2) = 9.80665d0
ONE (ZERO, TWO, HALF) HOLDS THE NUMBER ONE (ZERO, TWO, HALF)
PI HOLDS THE NUMBER PI = 3.141592653589793238462643383279D0
RGAS IS THE UNIVERSAL GAS CONSTANT (J/MOLE-K) = 8.31441D0
TCRIT HOLDS THE CRITICAL POINT OF WATER (K) = 647.244989D0
ZERO DC HOLDS THE TEMPERATURE IN KELVIN EQUAL TO ZERO DEGREES
CENTIGRADE (K) = 273.15D0
ONETRD = 1/3, TWOTRD = 2/3
SBCON = STEPFAN-BOLTZMANN CONSTANT (WATT/M**2-K**4) = 5.67D-8

COMMON /CONST/  BIGNMP, BIGNMM, SMLNMP, SMLNMM, RNDNUM, EXPMIN
               , EXPMAX, XLMNX , XLOGMX, GRAVTY, ONE , PI , RGAS
               , TCRIT , ZERO , ZERO DC, ONETRD, TWOTRD, SBCON , TWO
               , HALF

*-* INCLUDE RN1SIV
*-* COMMON/RN1SIV/RHONOM,ICOND,ICOEF,VISCOS,DENAIR,FREEMN,
      + CHI,GAMMA,FSLIP,STICK,PSAT,DELSAT,TURBDS,
      + TKGOP,DELDIF,DENSTY,FTHERM,VOLUME,
      + NREC,WTMOL,NB2A,NB2B,NB3,NB4,NGROW,
      + PGAS1,PGAS2,PGAS3,PGAS4,NCLS5,ICRLSE,NCLS8

*-* INCLUDE XMISC
*-* PARAMETER VALUES FOR XMISC ARRAY
*-* PARAMETER (KSSD=1, KGAS=2, KSTM=3, KWLV=4)

*-* COMMON/RN2SIV/RHONOM,ICOND,ICOEF,VISCOS,DENAIR,FREEMN,
      + CHI,GAMMA,FSLIP,STICK,PSAT,DELSAT,TURBDS,
      + TKGOP,DELDIF,DENSTY,FTHERM,VOLUME,
      + NREC,WTMOL,NB2A,NB2B,NB3,NB4,NGROW,
      + PGAS1,PGAS2,PGAS3,PGAS4,NCLS5,ICRLSE,NCLS8

*-* INCLUDE MXXEJ6
*-* COMMON /MXXEJ6/ IVOMIT
  DEBUGGING SWITCH = 0 NO DEBUGGING EDIT
  .NE. 0 THEN DEBUGGING EDIT

*-* COMMON/NCLS5/ICOND2

*-* COMMON/NCLS6/ICOND3

*-* COMMON/NCLS7/ICOND4

*-* COMMON/NCLS8/ICOND5

*---- IF HTO TRANSPORT, SUSPEND HTO BY BOILING

DO J=1,NCLS
   IF(IDCH(J).EQ.NCLS) THEN
      IF(FPCNAM(J)(1:3).EQ.'HTO') THEN
         GO TO 10
      ENDIF
   ENDIF
ENDDO
RETURN
K = NCLS
DO 13 N = 1, NNVOL
   IF(FCBOIL(N).LE.0.0) THEN
      C * * FLASHING
      FRACRL= MAX(-1.0,FCBOIL(N))
      DAERLN = FRACRL*AER1L(K,N)
      AER1L( K,N) = MAX(0.,AER1L( K,N)+DAERLN)
      AER1G(1,K,N) = MAX(0.,AER1G(1,K,N)-DAERLN)
   ELSE
C * * * CONDENSING
  K = NCLS
  DAERLN = 0.
  FRACRL = MIN( 1.0, FCBOIL(N))
  DO 14 J = 1, NSEC
     DAERGN = AER1G(J, K, N) * FRACRL
     DAERLN = DAERLN + DAERGN
     AER1G(J, K, N) = MAX(0., AER1G(J, K, N) - DAERGN)
  14 CONTINUE
  AER1L(K, N) = MAX(0., AER1L(K, N) + DAERLN)
ENDIF
  13 CONTINUE
C
C
RETURN
END
Appendix B

Code Listing for Oxidation Modifications
************ Changes made to Subroutine HSRUN2

C
C ******************************************************
C OXIDIZE SURFACES
C ******************************************************
C ******** WATER
C CALL OXIDIZW(NUMHS, NPTOT, NPMAX, NLOC, IGEOM, HSMULT,
1    STRU, IMAT, IBVL, ASURFL, BNDZL, IBVR,
2    ASURFR, BNDZR, HSL, HVL, HSR, HVR,
3    NUMVOL, NNVOL, NUMMAT, PRPRES, XMASS, CVSPC,
4    TEMPN, DELM, DELE, SPON, MATNMS, NUMNAM,
5    ENINP)
C
C IF(KO2.GE.0) THEN
C
C ******** AIR
C CALL OXIDIZA(NUMHS, NPTOT, NPMAX, NLOC, IGEOM, HSMULT,
1    XI, IMAT, IBVL, ASURFL, BNDZL, IBVR,
2    ASURFR, BNDZR, HSL, HVL, HSR, HVR,
3    NUMVOL, NNVOL, NUMMAT, PRPRES, XMASS, CVSPC,
4    TEMPN, DELM, DELE, SPON, MATNMS, NUMNAM,
5    ENINP)
ENDIF
ENDIF

******** New Subroutines

SUBROUTINE OXIDIZA( NUMHS, NPTOT, NPMAX, NLOC, IGEOM, HSMULT,
1    XI, IMAT, IBVL, ASURFL, BNDZL, IBVR,
2    ASURFR, BNDZR, HSL, HVL, HSR, HVR,
3    NUMVOL, NNVOL, NUMMAT, PRPRES, XMASS, CVSPC,
4    TEMPN, DELM, DELE, SPON, MATNMS, NUMNAM,
5    ENINP)
C
C
C THIS SUBROUTINE OXIDIZES BERYLILLUM ON HEAT STRUCTURES
C INPUT:
C
C NUMHS  - NUMBER OF HEAT STRUCTURES FOR THIS PROBLEM.
C NPTOT  - NUMBER OF TEMPERATURE NODES FOR ALL HEAT STRUCTURES.
C NPMAX  - MAXIMUM NUMBER OF TEMPERATURE NODES FOR ANY HEAT STRUCTURE.
C NLOC   - ARRAY CONTAINING LOCATION OF TEMPERATURE NODES AND MESH INTERVALS AT BOUNDARY SURFACES OF EACH HEAT STRUCTURE.
C IGEOM  - ARRAY CONTAINING GEOMETRY TYPE FOR EACH HEAT STRUCTURE.
C HSMULT - ARRAY CONTAINING MULTIPLICITY OF EACH HEAT STRUCTURE.
C XI     - ARRAY CONTAINING LOCATION OF EACH TEMPERATURE NODE FOR EACH HEAT STRUCTURE.
C IMAT   - ARRAY CONTAINING MATERIAL NUMBERS FOR MATERIALS IN MESH INTERVALS OF EACH HEAT STRUCTURE.
C IBVL   - ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS LEFT BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C ASURFL - ARRAY CONTAINING AREA OF LEFT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
C BNDZL  - ARRAY CONTAINING AXIAL LENGTH OF LEFT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
C IBVR   - ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS RIGHT BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C ASURFR - ARRAY CONTAINING AREA OF RIGHT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
C BNDZR  - ARRAY CONTAINING AXIAL LENGTH OF RIGHT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
C HSL    - ARRAY CONTAINING LEFT SURFACE WEIGHTS FOR EACH HEAT STRUCTURE.
C HVL    - ARRAY CONTAINING LEFT VOLUME WEIGHTS FOR EACH HEAT STRUCTURE.
C HSR    - ARRAY CONTAINING RIGHT SURFACE WEIGHTS FOR EACH HEAT STRUCTURE.
C HVR    - ARRAY CONTAINING RIGHT VOLUME WEIGHTS FOR EACH HEAT STRUCTURE.
C STRUCTURE.
C
C NUMVOL = NUMBER OF CONTROL VOLUMES PRESENT IN THIS PROBLEM.
C NNVOL = MAXIMUM OF NUMVOL AND ONE.
C NUMMAT = NUMBER OF MATERIALS PRESENT IN A CONTROL VOLUME.
C PRPRES = ARRAY CONTAINING PARTIAL PRESSURES OF ALL MATERIALS IN
C EACH CONTROL VOLUME.
C XMASS = ARRAY CONTAINING MASSES OF ALL MATERIALS IN EACH CONTROL
C VOLUME.
C CVSPC = ARRAY CONTAINING SPECIFIC THERMODYNAMIC PROPERTIES IN
C EACH CONTROL VOLUME.
C TEMPN = ARRAY CONTAINING TEMPERATURE DISTRIBUTION FOR EACH HEAT
C STRUCTURE AT NEW TIME.
C ENINP = ARRAY CONTAINING ENERGY INPUT BY INTERNAL AND SURFACE
C POWER SOURCES AND ENERGY TRANSFERRED FROM OTHER PACKAGES
C TO EACH HEAT STRUCTURE AT NEW TIME.
C DELM = ARRAY CONTAINING MASS CHANGES FOR ALL MATERIALS IN EACH
C CONTROL VOLUME DURING PRESENT COMPUTATIONAL CYCLE.
C DELE = ARRAY CONTAINING ENERGY CHANGES IN POOL, ATMOSPHERE, AND
C SURFACE OF EACH CONTROL VOLUME DURING PRESENT
C COMPUTATIONAL CYCLE.
C
C SCRATCH VARIABLES:
C
C VPOW = ARRAY CONTAINING VOLUMETRIC POWER WEIGHTS FOR EACH HEAT
C STRUCTURE.
C XIS = ARRAY CONTAINING OLD X-POINT
C
C ------------------------------------------------------------
C
C SAVE
C
C *- INCLUDE MXXTIM
C TIME COMMON BLOCK
C TIME = PROBLEM TIME
C DT = TIMESTEP
C CPU = CPU TIME
C NCYCLE = CYCLE NUMBER
C NDTPAK = PACKAGE CONTROLLING TIMESTEP
C COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
C COMMON /MXXTPK/ NDTPAK
C CHARACTER*3 NDTPAK
C
C *-
C
C *- INCLUDE CONST
C
C*************** PROCESSOR DEPENDENT NUMBERS
C BIGNMP (BIGNMM) = LARGEST POSITIVE (NEGATIVE) NUMBER
C SMLNMP (SMLNMM) = SMALLEST POSITIVE (NEGATIVE) NUMBER
C RNDNUM IS THE ROUNDOFF ASSOCIATED WITH THE NUMBER 1.
C EXPMIN (EXPMAX) = LARGEST NEGATIVE (POSITIVE) NUMBER THAT CAN
C BE USED AS AN ARGUMENT FOR THE EXP FUNCTION
C XLNMX (XLOGMX) = LARGEST NUMBER THAT CAN BE USED AS AN ARGUMENT
C FOR THE LN (LOG) FUNCTION
C****************** CONSTANTS
C GRAVTY IS THE ACCELERATION OF GRAVITY (M/S**2) = 9.80665D0
C ONE (ZERO, TWO, HALF) HOLDS THE NUMBER ONE (ZERO, TWO, HALF)
C PI HOLD THE NUMBER PI= 3.141592653589793238462643383279D0
C RGAS IS THE UNIVERSAL GAS CONSTANT (J/MOLE-K) = 8.31441D0
C TCRIT HOLDS THE CRITICAL POINT OF WATER (K) = 647.24989D0
C ZERODC HOLDS THE TEMPERATURE IN KELVIN EQUAL TO ZERO DEGREES
C CENTIGRADE (K) = 273.15D0
C ONETRD = 1/3, TWOTRD = 2/3
C SBCON = STEPFAN-BOLTZMANN CONSTANT (WATT/M**2-K**4) = 5.67D-8
C
C COMMON /CONST/ BIGNMP, BIGNMM, SMLNMP, SMLNMM, RNDNUM, EXPMIN
C 1 , EXPMAX, XLNMX, XLOGMX, GRAVTY, ONE , PI , RGAS
C 2 , TCRIT, ZERO, ZERODC, ONETRD, TWOTRD, SBCON , TWO
C 3 , HALF
*-  
C  -----------------------------------------------
C DIMENSION STATEMENTS FOR ARRAYS  
C  -----------------------------------------------
C
DIMENSION MATNMS(NUMNAM)
CHARACTER*8 MATNMS, MATNAM, PBUFR
CHARACTER*7 BLANK
C
DATA BLANK / '       ' /
C
DIMENSION  
C
1   IGEOM(NUMHS)  , HSMULT(NUMHS) , XI(NPTOT)     ,
2   IBVL(NUMHS)   , ASURFL(NUMHS) , BNDZL(NUMHS)  ,
3   IBVR(NUMHS)   , ASURFR(NUMHS) , BNDZR(NUMHS)  ,
4   HSL(NPTOT)    , HVL(NPTOT)    , HSR(NPTOT)    , HVR(NPTOT)
C
DIMENSION  
C
1   DELM(NUMMAT,NNVOL)            , DELE(3,NNVOL)                 ,
2   NLOC(4,NUMHS)                 , IMAT(NPTOT–NUMHS)             ,
3   PRPRES(NUMMAT,NNVOL)          , XMASS(NUMMAT,NNVOL)
C
DIMENSION  
C
1    XIS(10000)                      
C
DIMENSION  
C
1    SPW(2,NUMHS)                     , ENINP(2,NUMHS)
C
*- INCLUDE CVHSP  
C  THIS PARAMETER STATEMENT CONTAINS POINTERS TO THE SPECIFIC  
C  THERMODYNAMIC QUANTITIES IN ARRAYS TSPCN AND TSPCO.  
C  DIMENSION:  TSPCN(KSP1DM,NUMMAT,NUMVOL)
C
FIRST INDEX  
C        KSPRHO = MASS DENSITY (KG/M**3)
C        KSPH = SPECIFIC ENTHALPY (J/KG/K)
C        KSPE = SPECIFIC INTERNAL ENERGY (J/KG/K) (J/KG ?) SEE USE IN HSRUN2
C        KSP1DM = FIRST DIMENSION
C
SECOND INDEX  
C        MATERIAL NUMBER
C
THIRD INDEX  
C        VOLUME NUMBER
C
PARAMETER ( KSPRHO = 1 , KSPH = 2 , KSPE = 3 , KSP1DM = 3 )

*-  
C  -----------------------------------------------
C DIMENSION  
C
1    CVSPC(KSP1DM,NUMMAT,NNVOL)
C
*- INCLUDE NCGEOS  
C
*- NON-CONDENSIBLE GAS EOS COMMON BLOCKS  
C
PARAMETER (IEOS = 27)
C
COMMON /NCGNAM/ NAMGAS
C
COMMON /NCGEOS/ MAXGAS, NUMMAT, RNCG(IEOS), WN(IEOS), EMAXIM(IEOS),
2  KH2OP , KH2OLA , KH2OVA , KH2 , KO2 , KCO2 , KCO ,
3  KN2 , KNO , KN2O , KNH3 , KC2H2 , KCH4 , KC2H4 ,
4  KHE , KAR , KD2 ,

34
C LOGICAL REZONE, FIRST
C DATA NCYCLST /-1/
*
C
C ******************************************************
C ******************************************************
C ******************************************************
C --================================================================--------------------
C INITIALIZE FOLLOWING:
C 1. ERROR FLAGS FOR FALLBACK AND CONTROL VOLUME MASS IMBALANCE,
C 2. DELE AND DELM,
C 3. NUMBER OF MESH INTERVALS IN ALL HEAT STRUCTURES.
C --================================================================--------------------
C
C ***** RESTORE VARIABLE IN CASE OF BACKUP
IF(NCYCLE.NE.NCYCLST) THEN
IF(NPTOT.GT.10000) THEN
PRINT *,"NPTOT>10000-OXIDIZE"
STOP
ENDIF
DO 3003 N = 1, NPTOT
XIS(N) = XI(N)
3003 CONTINUE
ELSE
DO 3005 N = 1, NPTOT
XI(N) = XIS(N)
3005 CONTINUE
ENDIF
NCYCLST = NCYCLE
C
C --================================================================--------------------
C
C DO 5010 I = 1,NUMHS
IF (HSMULT(I) .EQ. ZERO) GOTO 5010
C
NODE1  = NLOC(1,I)
NODE2  = NLOC(2,I)
MESH1  = NLOC(3,I)
MESH2  = NLOC(4,I)
NMESH  = NODE2 - NODE1
NODES  = NMESH + 1
C
C --================================================================--------------------
C
C OBTAIN ORDER INDICES OF BOUNDARY VOLUMES
C --================================================================--------------------
C
IVL    = IBVL(I)
IVR    = IBVR(I)
C
C ******************
C BERYLLIUM OXIDATION OF STRUCTURE SURFACES
C
C ****** LEFT SURFACE
C
REZONE = .FALSE.
MNAME  = 3*(IMAT(MESH1)-1) + 1
MATNAM = MATNMS(MNAME)
IF((MATNAM(1:8).EQ.'BERYLLIU'
   .OR. (MATNAM(1:8).EQ.'CARBON  ' .OR. MATNAM(1:8).EQ.'CARBON--')
   .OR.  MATNAM(1:3).EQ.'CFC'
.OR.     MATNAM(1:8).EQ.'TUNGSTEN')
  .AND.              IVL.GT.0          ) THEN
  C
  THKMAT = 0
  NN     = NODE1
  FIRST  = .FALSE.
  VOID   = 1.0
  DO 3010 NM = MESH1, MESH2
       NN = NN + 1
       MNAMX  = 3*(IMAT(NM)-1) + 1
       MATNAM = MATNMS(MNAMX)
  IF(MATNAM(1:8).EQ.'BERYLLIU'
     .OR.(MATNAM(1:8).EQ.'CARBON ' .OR. MATNAM(1:8).EQ.'CARBON--')
     .OR.     MATNAM(1:3).EQ.'CFC'
     .OR.     MATNAM(1:8).EQ.'TUNGSTEN') THEN
       DELX  = XI(NN) - XI(NN-1)
       IF(DELX.GT.1.1E-6) THEN
          THKMAT = THKMAT + DELX
          IF(.NOT.FIRST) THEN
             FIRST  = .TRUE.
             MNAMX  = MNAMX + 1
             MATNAM = MATNMS(MNAMX)
          ENDIF
          IF(MATNAM(2:8) .NE. BLANK) THEN
             WRITE(PBUFF,*) MATNAM(2:8)
             READ (PBUFF,*) VOID
          ENDIF
       ENDIF
  ENDIF
  3010 CONTINUE

  DELMAT  = 0.0
  XMO2    = XMASS(KO2,IVL) + DELM(KO2,IVL)
  PO2     = PRPRES(KO2,IVL)

  CALL OXIRATA(MATNMS(MNAME), DT, TEMPN(NODE1), XMO2, ASURFL(I)
                 , PO2, VOID, OXMO2, OXMCO2, QOX, THKMAT, DELMAT
                 , HSMULT(I))

  O2ENER = 0.
  IF(KO2.GT.0) THEN
     DELM(KO2,IVL) = DELM(KO2,IVL) + OXMO2 *HSMULT(I)
  ENDIF
  ENDIF

  CO2ENER = 0.
  IF(KCO2.GT.0) THEN
     DELM(KCO2,IVL) = DELM(KCO2,IVL) + OXMCO2 *HSMULT(I)
  ENDIF
  ENDIF

  DELE(2,IVL) = DELE(2,IVL)
               + OXMO2*O2ENER*HSMULT(I)
               + OXMCO2*CO2ENER*HSMULT(I)

  SPOW(1,I)  = SPOW(1,I)  + QOX/ASURFL(I)
  ENINP(1,I) = ENINP(1,I) + QOX*DT

  NN = NODE1 - 1
  DO 3015 NM = MESH1, MESH2
       NN = NN + 1
       MNAMX  = 3*(IMAT(NM)-1) + 1
       MATNAM = MATNMS(MNAMX)
  IF(MATNAM(1:8).EQ.'BERYLLIU'
     .OR.(MATNAM(1:8).EQ.'CARBON ' .OR. MATNAM(1:8).EQ.'CARBON--')
     .OR.     MATNAM(1:3).EQ.'CFC'
     .OR.     MATNAM(1:8).EQ.'TUNGSTEN') THEN
       DELX  = MIN(DELMAT, XI(NN+1)-XI(NN)-1.E-6)
       DELX  = MAX(DELX , 0.)
  ENDIF
  GO TO 3015
  ENDIF
  DELMAT = DELMAT - DELX
  DO 3020 N = NODE1, NN
XI(N) = XI(N) + DELX

C REZONE = .TRUE.

C ***** RIGHT SURFACE

C

MNAME = 3*(IMAT(MESH2)-1) + 1
MATNAM = MATNMS(MNAME)

C IF( MATNAM(1:8).EQ.'BERYLLIUM' 
  .OR. (MATNAM(1:8).EQ.'CARBON' .OR. MATNAM(1:8).EQ.'CARBON--') 
  .OR. (MATNAM(1:8).EQ.'CFC' 
  .OR. (MATNAM(1:8).EQ.'TUNGSTEN') 
  .AND. IVR.GT.0 ) THEN
THKMAT = 0
NN = NODE2 + 1
FIRST = .FALSE.
VOID = 1.0
DO 3030 NM = MESH2, MESH1, -1
NN = NN - 1
MNAMX = 3*(IMAT(NM)-1) + 1
MATNAM = MATNMS(MNAMX)
IF( MATNAM(1:8).EQ.'BERYLLIUM' 
  .OR. (MATNAM(1:8).EQ.'CARBON' .OR. MATNAM(1:8).EQ.'CARBON--') 
  .OR. (MATNAM(1:8).EQ.'CFC' 
  .OR. (MATNAM(1:8).EQ.'TUNGSTEN') THEN
DELX = XI(NN) - XI(NN-1)
C
IF(DELX.GT.1.1E-6) THEN
THKMAT = THKMAT + DELX
IF(.NOT.FIRST) THEN
FIRST = .TRUE.
MNAMX = MNAMX + 1
MATNAM = MATNMS(MNAMX)
IF( MATNAM(1:8).EQ.'BERYLLIUM' 
  .OR. (MATNAM(1:8).EQ.'CARBON' .OR. MATNAM(1:8).EQ.'CARBON--') 
  .OR. (MATNAM(1:8).EQ.'CFC' 
  .OR. (MATNAM(1:8).EQ.'TUNGSTEN') THEN
DELMAT = 0.0
XMO2 = XMASS(KO2,IVR) + DELM(KO2,IVR)
PO2 = PRPRES(KO2,IVR)
C
CALL OXIRATA(MATNMS(MNAME), DT, TEMPN(NODE2), XMO2, ASURFR(I) 
  . , PO2, VOID, OXMO2, OXMCO2, QOX, THKMAT, DELMAT 
  . , HSMULT(I))
O2ENER = 0.
IF(KO2.GT.0) THEN
DELM(KO2,IVR) = DELM(KO2,IVR) + OXMO2 *HSMULT(I)
CALL NGCENG ( KO2, TEMPN(NODE2), O2ENER, O2ENTH)
ENDIF
CO2ENER = 0.
IF(KCO2.GT.0) THEN
DELM(KCO2,IVR) = DELM(KCO2,IVR) + OXMO2 *HSMULT(I)
CALL NGCENG ( KCO2, TEMPN(NODE2), CO2ENER, CO2ENTH)
ENDIF
DELE(2,IVR) = DELE(2,IVR) 
  + OXMO2*O2ENER*HSMULT(I) 
  + OXMCO2*CO2ENER*HSMULT(I)
C
SPOW(2,I) = SPOW(2,I) + QOX/ASURFR(I)
ENINP(1,I) = ENINP(1,I) + QOX*DT
C
NM = NODE2 + 1
DO 3035 NM = MESH2, MESH1, -1
NN = NN - 1
MNAMX = 3*(IMAT(NM)-1) + 1
MATNAM = MATNMS(MNAMX)
IF(. MATNAM(1:8).EQ.'BERYLLIUM' . .OR. MATNAM(1:8).EQ.'CARBON' . .OR. MATNAM(1:8).EQ.'CARBON--')
. .OR. MATNAM(1:3).EQ.'CFC'
. .OR. MATNAM(1:8).EQ.'TUNGSTEN') THEN
DELMX = MIN(DELMAT, XI(NN)-XI(NN-1)-1.E-6)
DELPX = MAX(DELMX, 0.)
IF(DELMAT.LE.0.0) GO TO 3035
DELMAT = DELMAT - DELX
DO 3040 N = NN, NODE2
XI(N) = XI(N) - DELX
3040 CONTINUE
ENDIF
3035 CONTINUE
C
REZONE = .TRUE.
ENDIF
C
IF(REZONE) THEN
DO 3050 N = NODE1, NODE2
IF(N.GT.NODE1) THEN
DELMX = XI(N) - XI(N-1)
ELSE
DELMX = 1.
ENDIF
IF(N.LT.NODE2) THEN
DELPX = XI(N+1) - XI(N)
ELSE
DELPX = 1.
ENDIF
GO TO (3051, 3052, 3053, 3054, 3054), IGEOM(I)
3051 HSL(N) = 1./DELMX
HVL(N) = DELMX/2.
HSR(N) = 1./DELPX
HVR(N) = DELXP/2.
GO TO 3050
3052 HSL(N) = 2.*PI*(XI(N)-DELMX/2.)/DELMX
HVL(N) = PI*(XI(N)**2-(XI(N)-DELMX/2.)**2)
HSR(N) = 2.*PI*(XI(N)+DELPX/2.)/DELPX
HVR(N) = PI*((XI(N)+DELPX/2.)**2-XI(N)**2)
GO TO 3050
3053 HSL(N) = 4.*PI*(XI(N)-DELMX/2.)**2/DELMX
HVL(N) = 4.*PI*(XI(N)**3-(XI(N)-DELMX/2.)**3)/3.
HSR(N) = 4.*PI*(XI(N)+DELPX/2.)**2/DELPX
HVR(N) = 4.*PI*((XI(N)+DELPX/2.)**3-XI(N)**3)/3.
GO TO 3050
3054 HSL(N) = 2.*PI*(XI(N)-DELMX/2.)**2/DELMX
HVL(N) = 2.*PI*(XI(N)**3-(XI(N)-DELMX/2.)**3)/3.
HSR(N) = 2.*PI*(XI(N)+DELPX/2.)**2/DELPX
HVR(N) = 2.*PI*((XI(N)+DELPX/2.)**3-XI(N)**3)/3.
C
3050 CONTINUE
HVL(NODE1) = 0.
HVR(NODE2) = 0.
GO TO (3061, 3062, 3063, 3064, 3064), IGEOM(I)
3061 HSL(NODE1) = 1.
HVR(NODE1) = 1.
GO TO 3060
3062 HSL(NODE1) = 2.*PI*XI(NODE1)
HVR(NODE2) = 2.*PI*XI(NODE2)
GO TO 3060
3063 HSL(NODE1) = 4.*PI*XI(NODE1)**2
HVR(NODE2) = 4.*PI*XI(NODE2)**2
GO TO 3060
3064 HSL(NODE1) = 2.*PI*XI(NODE1)**2
HVR(NODE2) = 2.*PI*XI(NODE2)**2
SUBROUTINE OXIDIZW( NUMHS, NPTOT, NPMAX, NLOC, IGEOM, HSMULT, XI, IMAT, IBVL, ASURFL, BNDZL, IBVR, ASURFR, BNDZR, HSL, HVL, HSR, HVR, NUMVOL, NNVOL, NUMMAT, PRPRES, XMASS, CVSPC, TEMPN, DELM, DELE, SPOW, MATNMS, NUMNAM, ENINP )

THIS SUBROUTINE OXIDIZES (BE,C,W) HEAT STRUCTURES WITH WATER

INPUT:

NUMHS - NUMBER OF HEAT STRUCTURES FOR THIS PROBLEM.
NPTOT - NUMBER OF TEMPERATURE NODES FOR ALL HEAT STRUCTURES.
NPMAX - MAXIMUM NUMBER OF TEMPERATURE NODES FOR ANY HEAT STRUCTURE.
NLOC - ARRAY CONTAINING LOCATION OF TEMPERATURE NODES AND MESH INTERVALS AT BOUNDARY SURFACES OF EACH HEAT STRUCTURE.
IGEOM - ARRAY CONTAINING GEOMETRY TYPE FOR EACH HEAT STRUCTURE.
HSMULT - ARRAY CONTAINING MULTIPLICITY OF EACH HEAT STRUCTURE.
XI - ARRAY CONTAINING LOCATION OF EACH TEMPERATURE NODE FOR EACH HEAT STRUCTURE.
IMAT - ARRAY CONTAINING MATERIAL NUMBERS FOR MATERIALS IN MESH INTERVALS OF EACH HEAT STRUCTURE.
IBVL - ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS LEFT BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
ASURFL - ARRAY CONTAINING AREA OF LEFT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
BNDZL - ARRAY CONTAINING AXIAL LENGTH OF LEFT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
IBVR - ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS RIGHT BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
ASURFR - ARRAY CONTAINING AREA OF RIGHT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
BNDZR - ARRAY CONTAINING AXIAL LENGTH OF RIGHT BOUNDARY SURFACE OF EACH HEAT STRUCTURE.
HSL - ARRAY CONTAINING LEFT SURFACE WEIGHTS FOR EACH HEAT STRUCTURE.
HVL - ARRAY CONTAINING LEFT VOLUME WEIGHTS FOR EACH HEAT STRUCTURE.
HSR - ARRAY CONTAINING RIGHT SURFACE WEIGHTS FOR EACH HEAT STRUCTURE.
HVR - ARRAY CONTAINING RIGHT VOLUME WEIGHTS FOR EACH HEAT STRUCTURE.
NUMVOL - NUMBER OF CONTROL VOLUMES PRESENT IN THIS PROBLEM.
NNVOL - MAXIMUM OF NUMVOL AND ONE.
NUMMAT - NUMBER OF MATERIALS PRESENT IN A CONTROL VOLUME.
PRPRES - ARRAY CONTAINING PARTIAL PRESSURES OF ALL MATERIALS IN EACH CONTROL VOLUME.
XMASS - ARRAY CONTAINING MASSES OF ALL MATERIALS IN EACH CONTROL VOLUME.
CVSPC - ARRAY CONTAINING SPECIFIC THERMODYNAMIC PROPERTIES IN EACH CONTROL VOLUME.
TEMPN - ARRAY CONTAINING TEMPERATURE DISTRIBUTION FOR EACH HEAT STRUCTURE AT NEW TIME.
ENINP - ARRAY CONTAINING ENERGY INPUT BY INTERNAL AND SURFACE POWER SOURCES AND ENERGY TRANSFERRED FROM OTHER PACKAGES TO EACH HEAT STRUCTURE AT NEW TIME.
DELM - ARRAY CONTAINING MASS CHANGES FOR ALL MATERIALS IN EACH CONTROL VOLUME DURING PRESENT COMPUTATIONAL CYCLE.
DELE - ARRAY CONTAINING ENERGY CHANGES IN POOL, ATMOSPHERE, AND SURFACE OF EACH CONTROL VOLUME DURING PRESENT COMPUTATIONAL CYCLE.
C SCRATCH VARIABLES:
C
C     VPW = ARRAY CONTAINING VOLUMETRIC POWER WEIGHTS FOR EACH HEAT
C              STRUCTURE.
XIS = ARRAY CONTAINING OLD X-POINT
C
C----------------------------------------------------------------------
SAVE
*- INCLUDE MXXTIM
C
TIME COMMON BLOCK
C TIME = PROBLEM TIME
C DT = TIMESTEP
C CPU = CPU TIME
C NCYCLE = CYCLE NUMBER
C NDTPAK = PACKAGE CONTROLLING TIMESTEP
COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
COMMON /MXXTPK/ NDTPAK
CHARACTER*3 NDTPAK
C
*- INCLUDE CONST
C
************** PROCESSOR DEPENDENT NUMBERS
C
BIGNMP (BIGNMM) = LARGEST POSITIVE (NEGATIVE) NUMBER
C SMLNMP (SMLNMM) = SMALLEST POSITIVE (NEGATIVE) NUMBER
C RNDNUM IS THE ROUNDOFF ASSOCIATED WITH THE NUMBER 1.
C EXPMIN (EXPMAX) = LARGEST NEGATIVE (POSITIVE) NUMBER THAT CAN
C BE USED AS AN ARGUMENT FOR THE EXP FUNCTION
C XLNMX (XLOGMX) = LARGEST NUMBER THAT CAN BE USED AS AN ARGUMENT
C FOR THE LN (LOG) FUNCTION
C********************* CONSTANTS
C
GRAVTY IS THE ACCELERATION OF GRAVITY (M/S**2) = 9.80665D0
C ONE (ZERO, TWO, HALF) HOLDS THE NUMBER ONE (ZERO, TWO, HALF)
C PI HOLD THE NUMBER PI= 3.141592653589793238462643383279D0
C RGAS IS THE UNIVERSAL GAS CONSTANT (J/MOLE-K) = 8.31441D0
C TCRIT HOLDS THE CRITICAL POINT OF WATER (K) = 647.244989D0
C ZERODC HOLDS THE TEMPERATURE IN KELVIN EQUAL TO ZERO DEGREES
C CENTIGRADE (K) = 273.15D0
C ONETRD = 1/3,   TWOTRD = 2/3
C SBCON = STEPFAN-BOLTZMANN CONSTANT (WATT/M**2-K**4) = 5.67D-8
C
COMMON /CONST/  BIGNMP, BIGNMM, SMLNMP, SMLNMM, RNDNUM, EXPMIN
1      , EXPMAX, XLNMX, XLOGMX, GRAVTY, ONE   , PI    , RGAS
2      , TCRIT, ZERO  , ZERODC, ONETRD, TWOTRD, SBCON , TWO
3      , HALF
C
C----------------------------------------------------------------------
DIMENSION STATEMENTS FOR ARRAYS
C----------------------------------------------------------------------
C
DIMENSION MATNMS(NUMNAM)
CHARACTER*8 MATNMS
C
DIMENSION
C 1  IGEOM(NUMHS) , HSMULT(NUMHS) , XI(NPTOT) ,
2  IBVL(NUMHS)  , ASURFL(NUMHS) , BNDZL(NUMHS) ,
3  IBVR(NUMHS)  , ASURFR(NUMHS) , BNDZR(NUMHS) ,
4  HSL(NPTOT)   , HVL(NPTOT)   , HSR(NPTOT)   , HVR(NPTOT)
C
DIMENSION
C 1  DELM(NUMMAT,NNVOL) , DELE(3,NNVOL) ,
2  NLOC(4,NUMHS)  , IMAT(NPTOT-NUMHS) ,
3  PRPRES(NUMMAT,NNVOL) , XMASS(NUMMAT,NNVOL)
DIMENSION
1  TEMPN(NPTOT)

DIMENSION
1  XIS(10000)

DIMENSION
1  SPOW(2,NUMHS) , ENINP(2,NUMHS)

* - INCLUDE CVHSP

THIS PARAMETER STATMENT CONTAINS POINTERS TO THE SPECIFIC
THERMODYNAMIC QUANTITIES IN ARRAYS TSPCN AND TSPCO.
DIMENSION: TSPCN(KSP1DM,NUMMAT,NUMVOL)

FIRST INDEX
KSPRHO = MASS DENSITY (KG/M^3)
KSPH  = SPECIFIC ENTHALPY (J/KG/K)
KSPE  = SPECIFIC INTERNAL ENERGY (J/KG/K) (J/KG ?) SEE USE IN HSRUN2
KSP1DM = FIRST DIMENSION

SECOND INDEX
MATERIAL NUMBER

THIRD INDEX
VOLUME NUMBER

PARAMETER ( KSPRHO = 1 , KSPH = 2 , KSPE = 3 , KSP1DM = 3 )

DIMENSION
1  CVSPC(KSP1DM,NUMMAT,NNVOL)

* - INCLUDE NCGEOS

NON-CONDENSIBLE GAS EOS COMMON BLOCKS

PARAMETER (IEOS = 27)

COMMON /NCGNAME/ NAMGAS

CHARACTER*8 NAMGAS(IEOS)

COMMON /NCGEOS/ MAXGAS, NUMMAT, RNCG(IEOS), WM(IEOS), EMAXIM(IEOS),
2  KH2OP , KH2OLA , KH2OVA , KH2  , KO2  , KCO2 , KCO ,
3  KN2  , KNO  , KN2O  , KNH3 , KC2H2 , KC2H4 , KC2H4 ,
4  KHE  , KAR  , K2D  ,
5  KGASA , KGASB , KGASC , KGASD , KGASE , KGASF , KGASG ,
6  KGASH , KGASI , KGASJ ,

SCRATCH STORAGE FOR EOS WORK
7  XNCGSA(IEOS), XNCGSB(IEOS), XNCGSC(IEOS), XNCGSD(IEOS),
8  XNCGSE(IEOS), XNCGSF(IEOS), XNCGSG(IEOS), XNCGSH(IEOS)

LOGICAL REZONE

DATA NCYCLST /-1/

******************************************************************************
** INITIALIZE FOLLOWING:
** 1. ERROR FLAGS FOR Fallback AND CONTROL VOLUME MASS IMBALANCE,
** 2. DELE AND DELM,
** 3. NUMBER OF MESH INTERVALS IN ALL HEAT STRUCTURES.
******************************************************************************
C ***** RESTORE VARIABLE IN CASE OF BACKUP
   IF(NCYCLE.NE.NCYCLST) THEN
      IF(NPTOT.GT.10000) THEN
         PRINT *, "NPTOT>10000-OXIDIZE"
         STOP
      ENDIF
      DO 3003 N = 1, NPTOT
         XIS(N) = XI(N)
      3003 CONTINUE
   ELSE
      DO 3005 N = 1, NPTOT
         XI(N) = XIS(N)
      3005 CONTINUE
   ENDIF
   NCYCLST = NCYCLE
C
C----------------------------------------------------------------------
C
   DO 5010 I = 1, NUMHS
      IF (HSMULT(I) .EQ. ZERO) GOTO 5010
C
      NODE1  = NLOC(1,I)
      NODE2  = NLOC(2,I)
      MESH1  = NLOC(3,I)
      MESH2  = NLOC(4,I)
      NMESH  = NODE2 - NODE1
      NODES  = NMESH + 1
C
C----------------------------------------------------------------------
C OBTAIN ORDER INDICES OF BOUNDARY VOLUMES
C----------------------------------------------------------------------
C
      IVL    = IBVL(I)
      IVR    = IBVR(I)
C
C
C ******************************
C            BERYLLIUM OXIDATION OF STRUCTURE SURFACES
C
C ***** LEFT SURFACE
C
   THKMAT = 0
   NN     = NODE1
   DO 3010 NM = MESH1, MESH2
      NN = NN + 1
      NMAMX = 3*(IMAT(NM)-1) + 1
      IF((MATNMS(NMAMX) .EQ. 'BERYLLIU' 
         OR. MATNMS(NMAMX) .EQ. 'CARBON' 
         OR. MATNMS(NMAMX) .EQ. 'CFC' 
         OR. MATNMS(NMAMX) .EQ. 'TUNGSTEN')
         .AND. IVL.GT.0 ) THEN
      THKMAT = THKMAT + 1
   3010 CONTINUE
   DELMAT = 0.0
   XMH2O  = XMASS(3,IVL) + DELM(3,IVL)
   PH2O   = PRPRES(3,IVL)
C
   CALL OXIRATW(MATNMS(NMAMX), DT, TEMPN(NODE1), XMH2O, ASURFL(I) 
                 PH2O, OXMH2O, OXMCO, QOX, THKMAT, DELMAT 
                 HSMULT(I))
C
   42
DELM(3,IVL) = DELM(3,IVL) + OXMH2O*HSMULT(I)

H2ENER = 0.
IF(KH2.GT.0) THEN
  DELM(KH2,IVL) = DELM(KH2,IVL) + OXMH2 *HSMULT(I)
  CALL NCGENR ( KH2, TEMPN(NODE1), H2ENER, H2ENTH)
ENDIF

COENER = 0.
IF(KCO.GT.0) THEN
  DELM(KCO,IVL) = DELM(KCO,IVL) + OXMCO *HSMULT(I)
  CALL NCGENR ( KCO, TEMPN(NODE1), COENER, COENTH)
ENDIF

DELE(2,IVL) = DELE(2,IVL)
  + CVSPC(KSPE,3,IVL)*OXMH2O*HSMULT(I)
  + OXMH2*H2ENER*HSMULT(I)
  + OXMCO*COENER*HSMULT(I)

SPW(1,I) = SPW(1,I) + QOX/ASURFL(I)
ENINP(1,I) = ENINP(1,I) + QOX*DT

NN = NODE1 - 1
DO 3015 NM = MESH1, MESH2
NN = NN + 1
MNAME = 3*(IMAT(NM)-1) + 1
IF(MATNMS(MNAME).EQ.'BERYLLIU'
  .OR.  MATNMS(MNAME).EQ.'CARBON'
  .OR.  MATNMS(MNAME).EQ.'CFC'
  .OR.  MATNMS(MNAME).EQ.'TUNGSTEN') THEN
  DELX = MIN(DELMAT, XI(NN+1)-XI(NN)-1.E-6)
  DELX = MAX(DELX, 0.)
  IF(DELMAT.LE.0.0) GO TO 3015
  DELMAT = DELMAT - DELX
  DO 3020 N = NODE1, NN
  XI(N) = XI(N) + DELX
  3020 CONTINUE
ENDIF
3015 CONTINUE

REZONE = .TRUE.
ENDIF

****** RIGHT SURFACE

MNAME = 3*(IMAT(MESH2)-1) + 1
IF(MATNMS(MNAME).EQ.'BERYLLIU'
  .OR.  MATNMS(MNAME).EQ.'CARBON'
  .OR.  MATNMS(MNAME).EQ.'CFC'
  .OR.  MATNMS(MNAME).EQ.'TUNGSTEN') THEN
  DELX = XI(NN) - XI(NN-1)
  IF(DELX.GT.1.1E-6) THKMAT = THKMAT + DELX
ENDIF

DELMAT = 0.0
XMH2O = XMASS(3,IVR) + DELM(3,IVR)
PH2O = PRPRES(3,IVR)

CALL OXIRATW(MATNMS(MNAME), DT, TEMPN(NODE2), XMH2O, ASURFR(I),
  PH2O, OXMH2, OXMH2O, OXCO, QOX, THKMAT, DELMAT,
  HSMULT(I))
DELM(KH2,IVR) = DELM(KH2,IVR) + OXMH2O*HSMULT(I)
CALL NCGENR ( KH2, TEMPN(NODE2), H2ENER, H2ENTH)
ENDIF

DELM(KCO,IVR) = DELM(KCO,IVR) + OXMCO*HSMULT(I)
CALL NCGENR ( KCO, TEMPN(NODE2), COENER, COENTH)
ENDIF

DELE(2,IVR) = DELE(2,IVR) + CVSPC(KSPE,3,IVR)*OXMH2O*HSMULT(I)
 . + OXMH2*H2ENER*HSMULT(I)
 . + OXMCO*COENER*HSMULT(I)
C
SPOW(2,I) = SPOW(2,I) + QOX/ASURFR(I)
ENINP(1,I) = ENINP(1,I) + QOX*DT
C
NN = NODE2 + 1
DO 3035 NM = MESH2, MESH1, -1
NN = NN - 1
MNAME = 3*(IMAT(NM)-1) + 1
IF(MATNMS(MNAME).EQ.'BERYLLIU'
 . OR. MATNMS(MNAME).EQ.'CARBON' 
 . OR. MATNMS(MNAME).EQ.'CFC' 
 . OR. MATNMS(MNAME).EQ.'TUNGSTEN') THEN
DElx = MIN(DELMAT, XI(NN)-XI(NN-1)-1.E-6)
DElx = MAX(DElx, 0.)
IF(DELMAT.LT.0.0) GO TO 3035
DELMAT = DELMAT - DElx
DO 3040 N = NN, NODE2
XI(N) = XI(N) - DElx
3040 CONTINUE
ENDIF
3035 CONTINUE
ENDIF
C
IF(REZONE) THEN
DO 3050 N = NODE1, NODE2
IF(N.GT.NODE1) THEN
DElxM = XI(N) - XI(N-1)
ELSE
DElxM = 1.
ENDIF
IF(N.LT.NODE2) THEN
DElxP = XI(N+1) - XI(N)
ELSE
DElxP = 1.
ENDIF
GO TO (3051, 3052, 3053, 3054, 3054), IGEOM(I)
3051 HSL(N) = 1./DElxM
HVL(N) = DElxM/2.
HSR(N) = 1./DElxP
HVR(N) = DElxP/2.
GO TO 3050
3052 HSL(N) = 2.*PI*(XI(N)-DElxM/2.)/DElxM
HVL(N) = PI*(XI(N)**2-(XI(N)-DElxM/2.)**2)
HSR(N) = 2.*PI*(XI(N)+DElxP/2.)/DElxP
HVR(N) = PI*((XI(N)+DElxP/2.)**2-XI(N)**2)
GO TO 3050
3053 HSL(N) = 4.*PI*(XI(N)-DElxM/2.)**2/DElxM
HVL(N) = 4.*PI*(XI(N)**3-(XI(N)-DElxM/2.)**3)/3.
HSR(N) = 4.*PI*(XI(N)+DElxP/2.)**2/DElxP
HVR(N) = 4.*PI*((XI(N)+DElxP/2.)**3-XI(N)**3)/3.
GO TO 3050
3054 HSL(N) = 2.*PI*(XI(N)-DElxM/2.)**2/DElxM
HVL(N) = 2.*PI*(XI(N)**3-(XI(N)-DELXM/2.)**3)/3.
HSR(N) = 2.*PI*(XI(N)+DELXP/2.)**2/DELXP
HVR(N) = 2.*PI*((XI(N)+DELXP/2.)**3-XI(N)**3)/3.

C 3050 CONTINUE
HVL(NODE1) = 0.
HVR(NODE2) = 0.
GO TO (3061, 3062, 3063, 3064, 3064), IGEOM(I)

3061 HSL(NODE1) = 1.
HSR(NODE2) = 1.
GO TO 3060

3062 HSL(NODE1) = 2.*PI*XI(NODE1)
HSR(NODE2) = 2.*PI*XI(NODE2)
GO TO 3060

3063 HSL(NODE1) = 4.*PI*XI(NODE1)**2
HSR(NODE2) = 4.*PI*XI(NODE2)**2
GO TO 3060

3064 HSL(NODE1) = 2.*PI*XI(NODE1)**2
HSR(NODE2) = 2.*PI*XI(NODE2)**2

C 3060 CONTINUE
ENDIF

5010 CONTINUE
C RETURN
END

SUBROUTINE OXIRATA(MNAME, DT, TSURF, XMO2, ASURF, PO2, VOID, OXMO2,
.                  OXMCO2, QOX, THICK, DELTHK, HSMULT)
C
C ***** THIS SUBROUTINE OXIDIZES BE, C, W IN AIR
C
C        INPUT:  TSURF - SURFACE TEMPERATURE OF OXIDIZING HEAT STRUCTURE
C                XMO2  - MASS OF O2 IN BOUNDARY VOLUME
C                ASURF - SURFACE AREA OF OXIDIZING SURFACE
C                THICK - THICKNESS OF OXIDIZED STRUCTURE
C                PO2   - PARTIAL PRESSURE OF O2 IN BOUNDARY VOLUME
C                MNAME - NAME OF MATERIAL BEING OXIDIZED
C                DT    - TIME STEP
C                HSMULT- HEAT STRUCTURE MULTIPLIER (NUMBER OF IDENTICAL HS)
C
C        OUTPUT: OXMO2  - MASS OF O2 CONSUMED
C                OXMCO2 - MASS OF CO2 PRODUCED
C                QOX    - HEAT FLUX
C                THICK  - NEW THICKNESS OF OXIDIZED STRUCTURE
C
C *****  LOCAL VARIABLES
C                TOX
C                PMO2
C
REAL MBE, MH2, MH2O, MBEO, MCO, MCA, MW, MWO3, MO2, MCO2
CHARACTER*8 MNAME

DATA RHOBE / 1850.   /
DATA RHOCA / 1800.   /
DATA RHOW  / 19350.  /
DATA MBE   / 9.0122  /
DATA MH2   / 2.0141  /
DATA MH2O  / 18.0135 /
DATA MBEO  / 25.0116 /
DATA MCO   / 28.0110 /
DATA MCA   / 12.0112 /
DATA MW    / 183.85  /
DATA MWO3  / 231.85  /
DATA MO2   / 31.9988 /
DATA MCO2  / 44.0100 /
DATA RCON  / 1.98    /
DATA SFBE1 / 5.0 /
DATA SFBE2 / 5.0 /
DATA SFCA  / 2.0 /
DATA SFW   / 2.0 /
OXMO2 = 0.
OXMC02 = 0.
QOX = 0.
IF(XMO2.EQ.0.) RETURN

IF(MNAME.EQ.'BERYLLIU') THEN

BERYLLIUM & OXYGEN

2BE + O2 --> 2BeO

* * * H = F + T*S (PAGE D-45, CRC, 60TH EDITION)
HFBEO IN J/(KG-CO2)
ROX IN (KG-C)/(M2*S) ; QOX IN W/M2

* * * PRESSURE DEPENDENCE
PMO2 = PO2/0.181E5

TOX = TSURF
TOX = MIN(2000.0, TOX )

ROX = SFBE1*4.833E5*EXP(-26200.0/TOX)
IF(TOX.GT.1073.0) ROX = SFBE1*34.83*EXP(-15900.0/TOX)
ROX = VOID*PMO2*ROX
HFBEO = (-144220.D0+0.83D0*TOX +.00046D0*TOX**2+2.D0*1.24D5/TOX)*(4187.D0/MBE0)

DELTHK = MAX(0.0, DT*ROX/(VOID*RHOBE))
IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
OXMO2 = -DT*ROX*(0.5*MO2/MBE)*HSMULT*ASURF
IF(ABS(OXMO2).GT.0.5*XMO2) ROX = 0.5*ROX*XMO2/ABS(OXMO2)
DELTHK = MAX(0.0, DT*ROX/(VOID*RHOBE))
OXMO2 = -DT*ROX*(0.5*MO2/MBE)*ASURF
QOX = -HFBEO*ROX*(MBE0/MBE)*ASURF

ENDIF

IF(MNAME.EQ.'CARBON ' .OR. MNAME.EQ.'CFC ') THEN

CARBON & OXYGEN

C + O2 --> CO2

* * * H = F + T*S (PAGE D-45, CRC, 60TH EDITION)
HFCO2 IN J/(KG-CO2)
ROX IN (KG-C)/(M2*S) ; QOX IN W/M2

* * * PRESSURE DEPENDENCE
PMO2 = PO2/0.181E5

TOX = TSURF
TOX = MIN(2000.0, TOX )

IF(MNAME.EQ.'CARBON ') THEN
ROX = 0.2472D0*EXP(-5710.0/TOX)
IF(TOX.GT.1273.D0) ROX = 1.56E-2*EXP(-2260.0/TOX)
ELSE
ROX = 3.6308E+01*EXP(-1.3475E+04/TOX)
ENDIF
IF(TOX.GT.1448.D0) ROX = 1.57D-2*EXP(-2260.0/TOX)
ENDIF

ROX = VOID*PMO2*SFCA*ROX
HFCO2 = (-93690.0-1.63*TOX/2.303 +.07E-3*TOX**2-2.*0.23E5/TOX)*(4187./MCO2)

DELTHK = MAX(0.0, DT*ROX/(VOID*RHOCA))
IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
OXM2 = -DT*ROX*(MO2/MCA)*HSMULT*ASURF
IF(ABS(OXM2).GT.0.5*XMO2) ROX = 0.5*ROX*XMO2/ABS(OXM2)
DELTHK = MAX(0.0, DT*ROX/(VOID*RHOCA))
OXM2 = -DT*ROX*(MO2/MCA)*HSMULT*ASURF
OXMCO2 = DT*ROX*(MCO2/MCA)*ASURF
QOX = -HFCO2*ROX*(MCO2/MCA)*ASURF
ENDIF

C

IF(MNAME.EQ.'TUNGSTEN') THEN
   W + 3/2 O2 -> WO3
*C ** H = F + T*S (PAGE D-50, CRC, 60TH EDITION)
   HFWO3 IN J/(KG-WO3)
   ROX IN (KG-W)/(M2*S) ; QOX IN W/M2
C ** PRESSURE DEPENDENCE
   PMO2 = SQRT(PO2/0.181E5)
   TOX = TSURF
   ROX = 1.62E6*EXP(-24000.0/TOX)
   IF(POX.GT.973.) ROX = 7.448*EXP(-12170.0/TOX)
   ROX = VOID*PMO2*SFW*ROX
   HFWO3 = (-201180.+2.92*TOX/2.303
                  +.00181*TOX**2-2.*0.300E5/TOX)*(4187./MWO3)
   DELTHK = MAX(0.0, DT*ROX/(VOID*RHOW))
   IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
   OXM2 = -DT*ROX*(1.5*MO2/MW)*HSMULT*ASURF
   IF(ABS(OXM2).GT.0.5*XMO2) ROX = 0.5*ROX*XMO2/ABS(OXM2)
   DELTHK = MAX(0.0, DT*ROX/(VOID*RHOW))
   OXM2 = -DT*ROX*(1.5*MO2/MW)*ASURF
   QOX = -HFWO3*ROX*(MWO3/MW)*ASURF
ENDIF

C
RETURN
END

SUBROUTINE OXIRATW(MNAME, DT, TSURF, XMH2O, ASURF, PH2O, OXMH2
                      OXMCO, OXMH2O, QOX, THICK, DELTHK, HSMULT)
C
***** THIS SUBROUTINE OXIDIZES BE, C, W IN WATER
C
INPUT: TSURF - SURFACE TEMPERATURE
XMH2O - MASS OF H2O IN BOUNDARY VOLUME
ASURF - SURFACE AREA
THICK - BERYLLIUM THICKNESS
PH2O - PARTIAL PRESSURE OF H2O IN BOUNDARY VOLUME
MNAME - NAME OF MATERIAL BEING OXIDIZED
DT - TIME STEP
HSMULT- HEAT STRUCTURE MULTIPLIER (NUMBER OF IDENTICAL HS)
C
OUTPUT: OXMH2 - MASS OF H2 PRODUCED
OXMCO - MASS OF CO PRODUCED
OXMH2O - MASS OF H2O CONSUMED
QOX - HEATFLUX
THICK - NEW BERYLLIUM THICKNESS
C
REAL MBE, MH2, MH2O, MBEO, MCO, MCA, MW, MWO3, MO2, MCO2
CHARACTER*8 MNAME
C
DATA RHOBE / 1850.   /
DATA RHOCA / 1800.   /
DATA RHOW  / 19350.  /
DATA MBE   / 9.0122  /
DATA MH2   / 2.0141  /
DATA MH2O  / 18.0135 /
DATA MBEO  / 25.0116 /

47
DATA MCO / 28.0110 /
DATA MCA / 12.0112 /
DATA MW / 183.85 /
DATA MWO3 / 231.85 /
DATA RCON / 1.98 /
DATA SFBE1 / 2.0 /
DATA SFBE2 / 2.0 /
DATA SFCA / 2.0 /
DATA SFW / 2.0 /

OXMH2 = 0.
OXMH2O = 0.
OXMCO = 0.
QOX = 0.
IF(XMH2O.EQ.0.) RETURN

IF(MNAME.EQ.'BERYLLIUM') THEN

C *** Beryllium/steam oxidation
C Be + H2O --> BeO + H2
C
C * * Get hydrogen production and reaction energy
C
C * * * Beryllium
C HFBEO in J/(kg-BeO) ; HFH20 in J/(kg-H2O)
C ROX in (kg-Be)/(m2*s) ; QOX in W/m2
C
C * * * PRESSURE DEPENDENCE
PMH2O = (PH2O/0.86E5)**0.9
C
TOX = TSURF
TOX = MIN(2000.0, TOX )
C
C * * * DENSE
ROX = SFBE1*0.1205D0*EXP(-13465.0/TOX)
IF(TOX.GT.773.) ROX = SFBE1*4.378E15*EXP(-42933.0/TOX)
C
C * * * POROUS
EQU1 = 49.398*EXP(-12500.0/TOX)
IF(TOX.LT.1133.) THEN
ROX2 = 2.571E07*EXP(-28789.0/TOX)
ROX = SFBE2*SQRT(EQU1*ROX2)
ELSE
ROX3 = 35.357E0*EXP(-13387.0/TOX)
ROX = SFBE2*SQRT(EQU1*ROX3)
ENDIF

ROX = PMH2O*ROX
HFBEO = (-144220.+1.91*TOX/2.303
+ .00046*TOX**2+2.*1.24e5/TOX)*(4187./MBEO)
HFH2O = (-56930.0-6.75*TOX/2.303
+ .00064*TOX**2-2.*0.08e5/TOX)*(4187./MH2O)
C
DELTHK = MAX(0.0, DT*ROX/RHOBE)
IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
OXMH20= -DT*ROX*(MH2O/MBE)*HSMULT*ASURF
DELTHK = MAX(0.0, DT*ROX/RHOBE)
OXMH20= -DT*ROX*(MH2O/MBE)*ASURF
DELTHK = MAX(0.0, DT*ROX/RHOBE)
OXMH20= -DT*ROX*(MH2O/MBE)*ASURF
QOX = -(HFBEO-HFH2O*(MH2O/MBE)))*ROX*(MBEO/MBE)*ASURF

ENDIF

ENDIF

IF(MNAME.EQ.'CARBON ' .OR. MNAME.EQ.'CFC ' ) THEN
CARBON & STEAM
C
C        C + H2O -> CO + H2
C
C  * * * H = F + T*S (PAGE D-45, CRC, 60TH EDITION)
C        HFCO  IN J/(KG-CO) ;  HFH2O IN J/(KG-H2O)
C        ROX IN (KG-C)/(M2*S) ;  QOX IN W/M2
C
C  * * * PRESSURE DEPENDENCE
C
P = PH2O/0.86E5
C
TOX = TSURF
ROX = MIN(2000.0, TOX )
ROX = 8.571E7*EXP(-70800./(RCON*TOX))
IF(TOX.GT.1470.) ROX = 5.196*EXP(-22500./(RCON*TOX))
HFCO = (-25400.0-2.05*TOX/2.303
    -.000027*TOX**2-2.*1.095E5/TOX)*(4187./MCO)
HFH2O = (-56930.0-6.75*TOX/2.303
    +.00064*TOX**2-2.*0.08e5/TOX)*(4187./MH2O)
C
DELTHK = MAX(0.0, DT*ROX/RHOCA)
IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
OXMH2O = -DT*ROX*(MH2O/MCA)*HSMULT*ASURF
IF(ABS(OXMH2O).GT.0.5*XMH2O) ROX = 0.5*ROX*XMH2O/ABS(OXMH2O)
DELTHK = MAX(0.0, DT*ROX/RHOCA)
OXMH2O = -DT*ROX*(MH2O/MCA)*ASURF
OXMH2 = DT*ROX*(MH2/MCA)*ASURF
OXMC0 = DT*ROX*(MCO/MCA)*ASURF
QOX = -(HFWO3-HFH2O*(3.*MH2O/MW))*(MWO3/MW)*ASURF
C
ENDIF
C
IF(MNAME.EQ.'TUNGSTEN') THEN
C
W + 3H2O -> WO3 + 3H2
C
C  * * * H = F + T*S (PAGE D-50, CRC, 60TH EDITION)
C        HFWO3 IN J/(KG-WO3) ;  HFH2O IN J/(KG-H2O)
C        ROX IN (KG-W)/(M2*S) ;  QOX IN W/M2
C
C  * * * PRESSURE DEPENDENCE
PM = PH2O/0.84E5**0.78
C
TOX = TSURF
ROX = PMH2O*SFW*41.230*EXP(-33106./(RCON*TOX))
HFWO3 = (-201180.+2.92*TOX/2.303
    +.00181*TOX**2-2.*0.300E5/TOX)*(4187./MWO3)
HFH2O = (-56930.0-6.75*TOX/2.303
    +.00064*TOX**2-2.*0.080E5/TOX)*(4187./MH2O)
C
DELTHK = MAX(0.0, DT*ROX/RHOCO)
IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
OXMH2O = -DT*ROX*(3.*MH2O/MW)*HSMULT*ASURF
IF(ABS(OXMH2O).GT.0.5*XMH2O) ROX = 0.5*ROX*XMH2O/ABS(OXMH2O)
DELTHK = MAX(0.0, DT*ROX/RHOCO)
OXMH2O = -DT*ROX*(3.*MH2O/MW)*ASURF
OXMH2 = DT*ROX*(3.*MH2/MW)*ASURF
QOX = -(HFWO3-HFH2O*(3.*MH2O/MW03))*(MWO3/MW)*ASURF
C
ENDIF
C
RETURN
END
Appendix C

Code Listing for Aerosol Transport Modifications
***** CHANGES TO SUBROUTINE RN1RN4

***** change to line 197
COMMON/RN1SIV/RHONDP,ICOND,ICOEF,VISCOS,DENAIR,FREEMN,

***** added after 283
LOGICAL GVONLY, FLGHTO, STDMEL
DATA STDMEL /.FALSE./

***** added after line 306
C
FLAGGV = 0.
IF(.NOT.STDMEL) THEN
IF(GVONLY) FLAGGV = 1.
C

***** changes to line 315
CALL RN1DSC(NSEC,DSEC,XMSEC,RHONDP,2)

***** changes to line 319
WRITE(8,3307) NSEC,DSEC(1),DSEC(1+NSEC),RHONDP

***** added after line 424
TMASS = 0.
H2OMS = 0.
DO I=1,NSEC
  H2OMS = H2OMS + AER1GN(I,14,N)
  DO K=1,NCLS
    TMASS = TMASS + AER1GN(I,K,N)
  ENDDO
ENDO
RHONOM = RHONDP
IF(.NOT.STDMEL) THEN
  IF(TMASS.GT.0.0) THEN
    XMFRAC = H2OMS/TMASS
    RHONOM = RLIQ*XMFRAC + RHONOM*(1.-XMFRAC)
  ENDF
ENDIF

***** added after line 555
IF(STDMEL) THEN
  GRAVDEP = COEFAV(L3M1+3)
ELSE
  GRAVDEP = FGRAV(TVOL(2,N),RHOBLK,VISGAS,WMGAS,DPART,RHONOM)
ENDIF
C
array TDBMAR(9,NVOL) (turbulent bend deposition array)
contains the following
index 1: flag for valid info:
-1=invalid entry
1=valid entry
index 2: actual volume id
index 3: characteristic dimension (i.e. pipe diameter)
index 4: number of bends in the bend model
index 5: turning angle of the bends (radians)
index 6: cross-sectional area over deposition area (bend deposition)
index 7: surface roughness for turbulent deposition model
index 8: surface film surface tension
index 9: surface film viscosity
BENDDEP = 0.0
IF(TDBMAR(4,N).GT.0.0 .AND. .NOT.STDMEL) THEN
SR = MAX(TDBMAR(7,N)-DEPHGT,DPMIN)
DELFLM = MAX(DEPHGT,0.0)
ACOVRAS = PI*TDBMAR(3,N)**2/(4.*VOLUME)
BENDDEP = FBEND( VGAS, DPART, VISGAS, RHONOM, TDBMAR(3,N)
.,
   TDBMAR(4,N), TDBMAR(5,N), ACOVRAS, TDBMAR(6,N)
.,
   RHOBLK, TDBMAR(8,N), TDBMAR(9,N), DELFLM)
BENDDEP = BENDDEP*(1.-FLAGGV)
ENDIF
TURBDEP = 0.0
IF(TDBMAR(7,N).GT.0.0 .AND. .NOT.STDMEL) THEN
SR = MAX(TDBMAR(7,N)-DEPHGT,DPMIN)
DELFLM = MAX(DEPHGT,0.0)
TURBDEP = FTURB( VGAS, DPART, VISGAS, RHONOM, TVOL(2,N), RHOBLK
.,
   TDBMAR(3,N), SR, WMGAS, TDBMAR(8,N), TDBMAR(9,N), DELFLM)
TURBDEP = TURBDEP*(1.-FLAGGV)
ENDIF
C
C * * THERMOPHORESIS COEFFICIENT
C
IF(STDMEL) THEN
THRMDEP = COEFAV(L3M1+2)
ELSE
THRMDEP = FTHRM(TVOL(2,N),RHOBLK,VISGAS,WMGAS,DPART)
ENDIF
THRMDEP = THRMDEP*(1.-FLAGGV)
C
C * * DIFFUSION
C
IF(STDMEL) THEN
DIFFDEP = COEFAV(L3M1+1)
ELSE
DIFFDEP = FDIFF(TVOL(2,N),RHOBLK,VISGAS,WMGAS,DPART)
ENDIF
DIFFDEP = DIFFDEP*(1.-FLAGGV)
C
****** added after 741

IF(STDMEL) THEN
GRAVDEP = COEFAV(ITERM+3)
ELSE
GRAVDEP = FGRAV(TVOL(2,N),RHOBLK,VISGAS,WMGAS,DPART,RHONOM)
ENDIF

array TDBMAR(9,NVOL) (turbulent bend deposition array)
contains the following
index 1: flag for valid info:
   -1=invalid entry
      1=valid entry
index 2: actual volume id
index 3: characteristic dimension (i.e. pipe diameter)
index 4: number of bends in the bend model
index 5: turning angle of the bends (radians)
index 6: cross-sectional area over deposition area (bend deposition)
index 7: surface roughness for turbulent deposition model
index 8: surface film surface tension
index 9: surface film viscosity
BENDDEP = 0.0
IF(TDBMAR(4,N).GT.0.0 .AND. .NOT.STDMEL) THEN
SR = MAX(TDBMAR(7,N)-DEPHGT,DPMIN)
DELFLM = MAX(DEPHGT,0.0)
ACOVRAS = PI*TDBMAR(3,N)**2/(4.*VOLUME)
BENDDEP = FBEND( VGAS, DPART, VISGAS, RHONOM, TDBMAR(3,N)
.,
   TDBMAR(4,N), TDBMAR(5,N), ACOVRAS, TDBMAR(6,N)
.,
   RHOBLK, TDBMAR(8,N), TDBMAR(9,N), DELFLM)
BENDDEP = BENDDEP*(1.-FLAGGV)
ENDIF
TURBDEP = 0.0
IF(TDBMAR(7,N).GT.0.0 .AND. .NOT.STDMEL) THEN
SR = MAX(TDBMAR(7,N)-DEPHGT,DPMIN)
DELFML = MAX(DEPHGT,0.0)
TURBDEP = FTURB(VGAS,DPART,VISGAS,RHONOM,TVO{2,N},RHOB{L},TDBMAR(3,N),SR,WMGAS,TDBMAR(8,N),TDBMAR(9,N),DELFML)
    TURBDEP = TURBDEP*(1.-FLAGGV)
ENDIF
C
C * * THERMOPHORESIS COEFFICIENT
C
IF(STDMEL) THEN
    THRMDEP = COEFAV(ITERM+2)
ELSE
    THRMDEP = FTHRM(TVO{2,N},RHOB{L},VISGAS,WMGAS,DPART)
ENDIF
THRMDEP = THRMDEP*(1.-FLAGGV)
C
C * * DIFFUSION
C
IF(STDMEL) THEN
    DIFFDEP = COEFAV(ITERM+1)
ELSE
    DIFFDEP = FDIFF(TVO{2,N},RHOB{L},VISGAS,WMGAS,DPART)
ENDIF
DIFFDEP = DIFFDEP*(1.-FLAGGV)
C
Appendix D

Code Listing for Air Condensation Model Modifications
****** added after line 664
C
PARAMETER (TMIN = 273.16000D0, TMAX = 647.24498D0)
C
****** added after line 675
C
C ****** RESTORE VARIABLE IN CASE OF BACKUP
IF(NCYCLE.NE.NCYCLST) THEN
IF(NUMVOL.GT.10000) THEN
PRINT *,"NUMVOL>10000-HSRUN2"
STOP
ENDIF
DO N = 1, NUMHS
XMCL02S(N) = XMCL02(N)
XMCLN2S(N) = XMCLN2(N)
XMCR02S(N) = XMCR02(N)
XMCRN2S(N) = XMCRN2(N)
ENDDO
ELSE
DO N = 1, NUMHS
XMCL02(N) = XMCL02S(N)
XMCLN2(N) = XMCLN2S(N)
XMCR02(N) = XMCR02S(N)
XMCRN2(N) = XMCRN2S(N)
ENDDO
ENDIF
NCYCLST = NCYCLE
C
****** added after 1076
C
QCNCL = ZERO
***** changes to line 1099

I                 XMFLL         , HTICEL        , VOLSFL        ,
J                 FLXMLN2       , FLXMLO2       , QCNCL         ,
K                 XMCLO2(I)     , XMCLN2(I)     ,
L                 HMCLO2(I)     , HMCLN2(I)     , IERR          )
C

****** added after line 1114
C
QCNCR = ZERO
C
****** changes to line 1133

I                 XMFLR         , HTICER        , VOLSFR        ,
J                 FLXMRN2       , FLXMR02       , QCNCR         ,
K                 XMCRO2(I)     , XMCRN2(I)     ,
L                 HMCRO2(I)     , HMCRN2(I)     , IERR          )

****** change made to line 1229

1       - AA*DBNDYP + (VPOW(MESH1)+QCNCL)*HVR(NODE1)*DTP

****** change made to line 1311

1       - CC*DBNDYP + (VPOW(MESH2)+QCNCR)*HVL(NODE2)*DTP

******** added after line 1650
C  * * ACCOUNT FOR O2 AND N2 CONDENSATION
C
IF(KO2.GE.0) THEN
CMO2 = FLXMLO2*ASURFL(I)*DT
IF(CMO2.GT.0.5*XMASS(KO2,IVL)) CMO2 = 0.5*XMASS(KO2,IVL)
XMCLO2(I) = MAX(0.,XMCLO2(I)+CMO2)
DELM(KO2,IVL) = DELM(KO2,IVL) - CMO2*HSMULT(I)
HFLO2 = CVSPC(KSPH,KO2,IVL) - HMCLO2(I)
IF(CMO2.GT.0.0) THEN
DELEAL = DELEAL - CVSPC(KSPH,KO2,IVL)*CMO2*HSMULT(I)
ELSE
DELEAL = DELEAL - HFLO2*CMO2*HSMULT(I)
ENDIF
HMCLO2(I) = CMO2 * HFLO2
ENDIF
C
IF(KN2.GE.0) THEN
CMN2 = FLXMLN2*ASURFL(I)*DT
IF(CMN2.GT.0.5*XMASS(KN2,IVL)) CMN2 = 0.5*XMASS(KN2,IVL)
XMCLN2(I) = MAX(0.,XMCLN2(I)+CMN2)
DELM(KN2,IVL) = DELM(KN2,IVL) - CMN2*HSMULT(I)
HFLN2 = CVSPC(KSPH,KN2,IVL) - HMCLN2(I)
IF(CMN2.GT.0.0) THEN
DELEAL = DELEAL - CVSPC(KSPH,KN2,IVL)*CMN2*HSMULT(I)
ELSE
DELEAL = DELEAL - HFLN2*CMN2*HSMULT(I)
ENDIF
HMCLN2(I) = CMN2 * HFLN2
ENDIF
C
****** added after line 1664
C
C  * * ACCOUNT FOR O2 AND N2 CONDENSATION
C
IF(KO2.GE.0) THEN
CMO2 = FLXMR02*ASURFR(I)*DT
IF(CMO2.GT.0.5*XMASS(KO2,IVR)) CMO2 = 0.5*XMASS(KO2,IVR)
XMCR02(I) = MAX(0.,XMCR02(I)+CMO2)
DELM(KO2,IVR) = DELM(KO2,IVR) - CM02*HSMULT(I)
HFRO2 = CVSPC(KSPH,KO2,IVR) - HMCR02(I)
IF(CMO2.GT.0.0) THEN
DELEAR = DELEAR - CVSPC(KSPH,KO2,IVR)*CMO2*HSMULT(I)
ELSE
DELEAR = DELEAR - HFRO2*CM02*HSMULT(I)
ENDIF
HMCR02(I) = CMO2 * HFRO2
ENDIF
C
IF(KN2.GE.0) THEN
CMN2 = FLXMRN2*ASURFR(I)*DT
IF(CMN2.GT.0.5*XMASS(KN2,IVR)) CMN2 = 0.5*XMASS(KN2,IVR)
XMCRN2(I) = MAX(0.,XMCRN2(I)+CMN2)
DELM(KN2,IVR) = DELM(KN2,IVR) - CMN2*HSMULT(I)
HFRN2 = CVSPC(KSPH,KN2,IVR) - HMCRN2(I)
IF(CMN2.GT.0.0) THEN
DELEAR = DELEAR - CVSPC(KSPH,KN2,IVR)*CMN2*HSMULT(I)
ELSE
DELEAR = DELEAR - HFRN2*CMN2*HSMULT(I)
ENDIF
HMCRN2(I) = CMN2 * HFRN2
ENDIF
C
********** CHANGES MADE TO SUBROUTINE HSENST

******* added after line 109
C
C * * COMMON FOR AIR CONDENSATION (B.J.M)
C
COMMON /AIRCON/
   . XMCL02(10000), XMCLN2(10000), XMCL02S(10000), XMCLN2S(10000)
   , XMCR02(10000), XMCRN2(10000), XMCR02S(10000), XMCRN2S(10000)
   , HMCL02(10000), HMCLN2(10000), HMCR02(10000), HMCRN2(10000)
*-

******* added after line 156
C * * ACCOUNT FOR CHANGE FOR POSSIBLE N2 AND O2 FILMS
      3 + HMCL02(I) + HMCLN2(I) + HMCR02(I) + HMCRN2(I)

********** CHANGES TO SUBROUTINE HSTRAN

***** changes to line 10

9          XMFLST, HTCICE, VOLSF , XMFLXN2,XMFLXO2,QCNC ,
A      XMCO2 , XMCN2 , HMCO2 , HMCN2 , IERR)

***** inserted after line 418
C
C * * * CONSIDER N2 AND O2 CONDENSATION
C
XMFLXN2 = ZERO
XMFLXO2 = ZERO
QCNC = ZERO
HMCN2 = ZERO
HMCO2 = ZERO
C
IF(IATMS.GT.0.0) THEN
PO2 = ZERO
IF(KO2.GT.0) PO2 = PRPRES(KO2)
FN2 = ZERO
IF(KN2.GT.0) PN2 = PRPRES(KN2)
C
IF((XMCN2.GT.ZERO .OR. PN2.GT.ZERO) .AND. TSURF.LT.126.2) THEN
  ICNCRN = 3
  TAVE = 0.5*(TMTCT+TATMS)
  CALL MPEVAL(NUMDIF,XMASS,ICNCRN,KN2,NUMMAT,'CVH',
              NUMMAT,PRES,TAVE,DAB,IGERR)
  IF ( IGERR .NE. 0 ) THEN
    IERR = 1
    DAB = ZERO
    CALL MESERA('*** MP CALL FROM HSTRAN DAB',' ',' ',0)
  END IF
  CALL CONDNC('N2 ', PRES, TATMS, TMTCT, DAB, PO2, PN2,
               VISC, XNU, RHO, PR, CLN, XMFLXN2, HFG, RHOSL)
  CMN2 = XMFLXN2*ASURF*DT
  IF(CMN2.GT.0.5*XMASS(KN2)) CMN2 = 0.5*XMASS(KN2)
  XMFLXN2 = MAX(0.,XMFLXN2+CMN2)
  QSNC = HFG*CMN2*HSMULT
  IF(TSURF.GT.63.15) THEN
    FLMTHK = XMFLXN2/RHOSL/ASURF
    IF(FLMTHK.GT.FILMAX) CMN2 = (FLMAXM-FLMTHK)*RHOSL*ASURF
  ENDIF
  XMFLXN2 = CMN2/ASURF/DT
  HMNCN2 = HFG
ENDIF
C
IF((XMCO2.GT.ZERO .OR. PO2.GT.ZERO) .AND. TSURF.LT.154.58) THEN
  ICNCRN = 3
  TAVE = 0.5*(TMTCT+TATMS)
  CALL MPEVAL(NUMDIF,XMASS,ICNCRN,KO2,NUMMAT,'CVH',
              NUMMAT,PRES,TAVE,DAB,IGERR)
  IF ( IGERR .NE. 0 ) THEN
    IERR = 1
    DAB = ZERO
    CALL MESERA('*** MP CALL FROM HSTRAN DAB',' ',' ',0)
  END IF
  CALL CONDNC('O2 ', PRES, TATMS, TMTCT, DAB, PO2, PN2,
               VISC, XNU, RHO, PR, CLN, XMFLXO2, HFG, RHOSL)
  CMO2 = XMFLXO2*ASURF*DT
  IF(CMO2.GT.0.5*XMASS(KO2)) CMO2 = 0.5*XMASS(KO2)
  XMFLXO2 = MAX(0.,XMFLXO2+CMO2)
  QCNC = QCNC + HFG*CMO2*HSMULT
  IF(TSURF.GT.54.34) THEN
    FLMTHK = XMFLXO2/RHOSL/ASURF
    IF(FLMTHK.GT.FILMAX) CMO2 = (FLMAXM-FLMTHK)*RHOSL*ASURF
  ENDIF
  XMFLXO2 = CMO2/ASURF/DT
  HMCO2 = HFG
  QCNC  = QCNC/VOLSF/DT
ENDIF
C
ENDIF
C
******** CHANGES MADE TO MPVIS

****** changes to lines 142 and 143
C
C * WILKE & LEE FORMULA - PG 587 OF REID, PROPERTIES OF GASES & LIQUIDS
C
SQRMB = SQRT(2./(PPT001/XMOLW1+PPT001/XMOLW2))

58
PCDIF  = (3.03-0.98/SQRMAB)*1.01325E-2  
CEDIFF = PCDIF*TEMP**1.5/(PRESS*SQRMAB*SIG*SIG*OMEGA)  
CEDIFF = PCDIF*SQRT(TEMP**3*(PPT001/XMOLW1+PPT001/XMOLW2))/1  
(PRESS*SIG*SIG*OMEGA)  

********** CHANGES MADE TO MXXRS

***** added after line 153

C * * COMMON FOR AIR CONDENSATION (BJM)
C COMMON /AIRCON/
.   XMCL02(10000), XMCLN2(10000), XMCL02S(10000), XMCLN2S(10000)
.   XMCR02(10000), XMCRN2(10000), XMCR02S(10000), XMCRN2S(10000)
.   HMCL02(10000), HMCN02(10000), HMCN02S(10000)
.*  PARAMETER (NWCON = 12*10000)

***** added after line 175

C * * COMMON FOR AIR CONDENSATION (BJM)
C WRITE (NRESX) (XMCL02(I),I=1,NWCON)

***** added after line 373

C * * COMMON FOR AIR CONDENSATION (BJM)
C READ (NRESX, END=192, ERR=192) (XMCL02(I),I=1,NWCON)

********** CHANGES MADE TO HSEDT

***** added after line 414

C * * COMMON FOR AIR CONDENSATION (BJM)
C COMMON /AIRCON/
.   XMCL02(10000), XMCLN2(10000), XMCL02S(10000), XMCLN2S(10000)
.   XMCR02(10000), XMCRN2(10000), XMCR02S(10000), XMCRN2S(10000)
.   HMCL02(10000), HMCN02(10000), HMCN02S(10000)
.*  CONDENSED O2 MASS HEAT STRUCTURE
DATA NAMES(46),NAMUNT(46)/*HS-CMO2-L','KG*/
DATA NAMES(47),NAMUNT(47)/*HS-CMO2-R','KG*/
C CONDENSED N2 MASS HEAT STRUCTURE
DATA NAMES(48),NAMUNT(48)/*HS-CMN2-L','KG*/
DATA NAMES(49),NAMUNT(49)/*HS-CMN2-R','KG*/

***** added after line 636

C * * COMMON FOR AIR CONDENSATION (BJM)
C LEFT CONDENSED O2 MASS
C CALL MXXPLK (NUMHS , NAMES(46) , NAMUNT(46), IHSNUM)
C RIGHT CONDENSED O2 MASS
C
CALL MXSPLK (NUMHS , NAMES(47) , NAMUNT(47), IHSNUM)
C
C LEFT CONDENSED N2 MASS
C
CALL MXSPLK (NUMHS , NAMES(48) , NAMUNT(48), IHSNUM)
C
C RIGHT CONDENSED N2 MASS
C
CALL MXSPLK (NUMHS , NAMES(49) , NAMUNT(49), IHSNUM)

***** added after line 961
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
C LEFT CONDENSED O2 MASS
C
DO I = 1,NUMHS
   SCDATA(I) = XMCLO2(I) * HSMULT(I)
ENDDO
C
CALL MXSPLW (PAK   , NUMHS , SCDATA)
C
C RIGHT CONDENSED O2 MASS
C
DO I = 1,NUMHS
   SCDATA(I) = XMCRO2(I) * HSMULT(I)
ENDDO
C
CALL MXSPLW (PAK   , NUMHS , SCDATA)
C
C LEFT CONDENSED N2 MASS
C
DO I = 1,NUMHS
   SCDATA(I) = XMCLN2(I) * HSMULT(I)
ENDDO
C
CALL MXSPLW (PAK   , NUMHS , SCDATA)
C
C RIGHT CONDENSED N2 MASS
C
DO I = 1,NUMHS
   SCDATA(I) = XMCRN2(I) * HSMULT(I)
ENDDO
C
CALL MXSPLW (PAK   , NUMHS , SCDATA)

***** added after line 1166
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
IF(XMCLO2(J).GT.0.0 .OR. XMCLN2(J).GT.0.0) THEN
   WRITE (NOUT,4741) XMCLO2(J),XMCLN2(J)
ENDIF

***** added after line 1187
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
IF(XMCRO2(J).GT.0.0 .OR. XMCRN2(J).GT.0.0) THEN
   WRITE (NOUT,4741) XMCRO2(J),XMCRN2(J)
ENDIF

***** added after line 1345

4741   FORMAT(/'***** CONDENSED O2 MASS(KG)=',1PE11.4,
               '***** CONDENSED N2 MASS(KG)=',1PE11.4/)
***** added after line 1524

* * COMMON FOR AIR CONDENSATION (BJM)

IF(XMCLO2(J).GT.0.0 .OR. XMCLN2(J).GT.0.0) THEN
  WRITE (NOUT,4741) XMCLO2(J),XMCLN2(J)
ENDIF

***** added after line 1545

* * COMMON FOR AIR CONDENSATION (BJM)

IF(XMCRO2(J).GT.0.0 .OR. XMCRN2(J).GT.0.0) THEN
  WRITE (NOUT,4741) XMCRO2(J),XMCRN2(J)
ENDIF

******* NEW SUBROUTINES

SUBROUTINE CONDNC(GASNAME, PRES, TEMP, TSURF, DAB, PO2, PN2
                   , VISC, XNU, RHO, PR, CLN, XMFLXNC, HFG, RHOSL)

C     THIS SUBROUTINE CALCULATES THE CONDENSATION RATES OF O2
C         AND N2 - GASES WHICH ARE NORMALLY TREATED AS NON-CONDENSIBLES
C         BY THE MELCOR CODE. THE CONDENSATION RATE IS DETERMINE AS
C         DOES MELCOR FOR WATER.
C
CHARACTER*3 GASNAME

COMMON/WCRCOEF/ FCO2(9), FCN2(9), DCO2(6), DCN2(6)
., ALPHAO2, TMAXO2, PMAXO2, TMINO2
., ALPHAN2, TMAXN2, PMAXN2, TMINN2
., CPGO2, CPGN2, HFSO2(22), HFSN2(15)
., TO2I(22), HFGO2(22), TN2I(15), HFGN2(15)
REAL*8 FCO2, FCN2, DCO2, DCN2
C
TSURFX = TSURF
IF(GASNAME.EQ.'O2 ') THEN
  IF(TSURFX.GT.TMAXO2) TSURFX = TMAXO2
  IF(TSURFX.LT.TMINO2) TSURFX = TMINO2
  DPATMS = MAX(1., PRES-PO2)
  TAVE = 0.5*(TSURFX+TEMP)
  CALL RHOVNC(GASNAME,TSURF,1)
  RHOSL = PFUND2(TSURFX,DCO2,TMAXO2)
  CPG = CPGO2
ENDIF
IF(GASNAME.EQ.'N2 ') THEN
  IF(TSURFX.GT.TMAXN2) TSURFX = TMAXN2
  IF(TSURFX.LT.TMINN2) TSURFX = TMINN2
  DPATMS = MAX(1., PRES-PN2)
  TAVE = 0.5*(TSURFX+TEMP)
  CALL RHOVNC(GASNAME,TSURF,1)
  RHOSL = PFUND2(TSURFX,DCN2,TMAXN2)
  CPG = CPGN2
ENDIF
C
SC = VISC/(RHO*DAB)
SH = XNU*SC**0.333/PR**0.3333
C
CALL SATNC(GASNAME,TSURF,PSURF,1)
CALL HFGNC(GASNAME,TSURF,1)
HFG = CPG * (TEMP - TSURF) + HFGS
XMNC = DAB*SH/CLN
DPSURF = MAX(1., PRES-PSURF)
XMFLXNC = XMNC*RHOV*ALOG(DPSURF/DPATMS)*EXP(-3000./PRES)
RETURN
END

SUBROUTINE HFGNC(GASNAME, TSAT, HFG, IMOD)
C
C       THIS SUBROUTINE CALCULATES HEAT OF VAPORIZATION
C          THE EQUATIONS USED ARE THOSE
C          FOR O2 AND N2 FOUND IN W.C. REYNOLDS, "THERMODYNAMIC
C          PROPERTIES IN SI"
C
CHARACTER*3 GASNAME
C
COMMON/WCRCOEF/ FCO2(9), FCN2(9), DCO2(6), DCN2(6)
   ., ALPHAO2, TMAXO2, PMAXO2, TMINO2
   ., ALPHAN2, TMAXN2, PMAXN2, TMINN2
   ., CPGO2 , CPGN2 , HFSO2(22), HFSN2(15)
   ., TO2I(22), HFGO2(22), TN2I(15), HFGN2(15)
REAL*8 FCO2, FCN2, DCO2, DCN2
DATA BUMP / 1.E-3 /
C
C * * * IMOD=1
IF(IMOD.EQ.1) THEN
  IF(GASNAME.EQ.'O2 ') THEN
    IF(TSAT.LT.TO2I(1)) THEN
      HFG  = HFGO2(1)*1000.
    ELSE
      DO I=2,22
        IM1 = I-1
        FRAC = (TSAT-TO2I(IM1))/(TO2I(I)-TO2I(IM1))
        IF(FRAC.GE.0.0 .AND. FRAC.LE.1.0) THEN
          HFG  = (HFGO2(IM1) + FRAC*(HFGO2(I)-HFGO2(IM1)))*1000.
        ELSE
          HFG = 0.
        ENDIF
      ENDDO
      RETURN
    ENDIF
    HFG = 0.
    ENDIF
  ENDIF
  IF(GASNAME.EQ.'N2 ') THEN
    IF(TSAT.LT.TN2I(1)) THEN
      HFG  = HFGN2(1)*1000.
    ELSE
      DO I=2,15
        IM1 = I-1
        FRAC = (TSAT-TN2I(IM1))/(TN2I(I)-TN2I(IM1))
        IF(FRAC.GE.0.0 .AND. FRAC.LE.1.0) THEN
          HFG  = (HFGN2(IM1) + FRAC*(HFGN2(I)-HFGN2(IM1)))*1000.
        ELSE
          HFG = 0.
        ENDIF
      ENDDO
      RETURN
    ENDIF
  ENDIF
  C * * * IMOD=2
ELSE
  TSATP = (1.+BUMP)*TSAT
  TSATM = (1.-BUMP)*TSAT
  IF(GASNAME.EQ.'O2 ') THEN
    TSATP = MIN(TSATP,TMAXO2)
    TSATM = MAX(TSATM,TMINO2)
    PSATP = PFUNS4(TSATP,FCO2,ALPHAO2,TMAXO2)
    PSATM = PFUNS4(TSATM,FCO2,ALPHAO2,TMAXO2)
    DPDT  = (PSATP-PSATM)/(TSATP-TSATM)
  ELSE
  ENDIF
  IF(GASNAME.EQ.'N2 ') THEN
    TSATP = MIN(TSATP,TMAXN2)
    TSATM = MAX(TSATM,TMINN2)
    PSATP = PFUNS4(TSATP,FCN2,ALPHAN2,TMAXN2)
    PSATM = PFUNS4(TSATM,FCN2,ALPHAN2,TMAXN2)
    DPDT  = (PSATP-PSATM)/(TSATP-TSATM)
  ELSE
  ENDIF
ENDIF
RETURN
END
RHOSL = PFUND2(TSAT, DCO2, TMAXO2)
ENDIF

C
IF(GASNAME.EQ.'N2') THEN
TSATP = MIN(TSATP, TMAXN2)
TSATM = MAX(TSATM, TMINN2)
PSATP = PFUN54(TSATP, FCN2, ALPHAN2, TMAXN2)
PSATM = PFUN54(TSATM, FCN2, ALPHAN2, TMAXN2)
DPDT = (PSATP-PSATM)/(TSATP-TSATM)
RHOSL = PFUND2(TSAT, DCN2, TMAXN2)
ENDIF

PSAT = 0.5*(PSATP+PSATM)
CALL RHOVNC(GASNAME, TSAT, PSAT, RHOSV)
VFG = MAX(0., 1./RHOSV - 1./RHOSL)
HFG = TSAT*VFG*DPDT
ENDIF

C
RETURN
END

BLOCK DATA REYCOEF

C

C * * PROPERTY INFORMATION FROM W.C. REYNOLDS

C

COMMON/WCRCOEF/ FCO2(9), FCN2(9), DCO2(6), DCN2(6), ALPHAO2, TMAXO2, PMAXO2, TMINO2, CPGO2, T02I(22), HFGO2(22), TN2I(15), HFGN2(15)
REAL*8 FCO2, FCN2, DCO2, D CN2, ALPHAO2, TMAXO2, PMAXO2, TMINO2, CPGO2, T02I(22), HFGO2(22), TN2I(15), HFGN2(15)

DATA FCO2 /
. -5.5819320390D+02, -1.0966262185D+02, -8.3456211630D-02
. 2.6603644330D-03, 1.6875023830D-05, -2.1262477120D-07
. 9.5741096780D-10, -1.6617640450D-12, 2.7545605710D+01 /
DATA ALPHAO2 / 1.91576 /
DATA TMAXO2 / 154.581 /
DATA PMAXO2 / 5.043E6 /
DATA TMINO2 / 54.34 /
DATA CPGO2 / 916.9 /

DATA FCN2 /
. 8.3944094440D+03, -1.8785191705D+03, -7.2822291650D+00
. 1.0228509660D-02, -1.6756023830D-05, -2.1262477120D-07
. 9.5741096780D-10, -1.6617640450D-12, 2.7545605710D+01 /
DATA ALPHAN2 / 1.95 /
DATA TMAXN2 / 126.2 /
DATA PMAXN2 / 3.4E6 /
DATA THMN2 / 54.34 /
DATA CPGN2 / 1038.3 /

DATA DCO2 /
. 4.3615175D+02, 7.5897189D+02, -4.2576866D+02
. 2.3487106D+03, -3.0474660D+03, 1.4850169D+03 /
DATA D CN2 /
. 3.1402991D+02, 4.4111015D+02, 9.4622994D+02
. -2.9067111D+03, 4.4785979D+03, -2.746914D+03 /

DATA TO2I /
. 54.34, 60., 65., 70., 75., 80., 85., 90., 90.19, 95.
. 150., 154.58 /
DATA HFGO2 /
. 242.37, 238.26, 234.43, 230.50, 226.47, 222.30, 217.92
. 213.23, 213.03, 208.14, 202.57, 196.45, 189.69, 182.17
. 173.75, 164.24, 153.30, 140.50, 125.11, 105.93, 79.53
. 0.0 /
DATA HFGN2 /
. 0.0, 9.23, 17.54, 25.88, 34.22, 42.56, 50.92
. 59.36, 59.69, 67.91, 76.61, 85.47, 94.52, 103.79
. 113.34, 123.24, 133.65, 144.77, 156.91, 170.52, 186.90
SUBROUTINE SATNC(GASNAME, TSAT, PSAT, IMOD)

CHARACTER*3 GASNAME

COMMON/WCRCOEF/ FCO2(9), FCN2(9), DCO2(6), DCN2(6)
    ALPHAO2, TMAXO2, PMAXO2, TMINO2
    ALPHAN2, TMAXN2, PMAXN2, TMINN2
    CPGO2, CPGN2, HFSO2(22), HFSN2(15)
    TO2I(22), HFGO2(22), TN2I(15), HFGN2(15)

REAL*8 FCO2, FCN2, DCO2, DCN2

IF(IMOD.EQ.1) THEN
    IF(GASNAME.EQ.'O2') THEN
        TSATX = MAX(TSAT, TMINO2)
        PSAT = PFUNS4(TSATX, FCO2, ALPHAO2, TMAXO2)
        IF(TSAT.LT.TSATX) PSAT = PSAT*TSAT/TSATX
    ENDIF
    IF(GASNAME.EQ.'N2') THEN
        TSATX = MAX(TSAT, TMINN2)
        PSAT = PFUNS4(TSATX, FCN2, ALPHAN2, TMAXN2)
        IF(TSAT.LT.TSATX) PSAT = PSAT*TSAT/TSATX
    ENDIF
ELSE
    IF(GASNAME.EQ.'N2') THEN
        TSAT = 0.5*(TMAXN2 + TMINN2)
        IF(PSAT.GT.PMAXN2) THEN
            TSAT = TMAXN2
            RETURN
        ENDIF
        PSMIN = PFUNS4(TMINN2, FCN2, ALPHAN2, TMAXN2)
        IF(PSAT.LT.PSMIN) THEN
            TSAT = TMINN2
            RETURN
        ENDIF
    ENDIF
    IF(GASNAME.EQ.'O2') THEN
        TSAT = 0.5*(TMAXO2 + TMINO2)
        IF(PSAT.GT.PMAXO2) THEN
            TSAT = TMAXO2
            RETURN
        ENDIF
        PSMIN = PFUNS4(TMINO2, FCO2, ALPHAO2, TMAXO2)
        IF(PSAT.LT.PSMIN) THEN
            TSAT = TMINO2
            RETURN
        ENDIF
ENDIF
C
EXTRP = 0.5
DO 10 NLOOP = 1, 100
C
TSATB = 1.0001*TSAT
IF(GASNAME.EQ.'O2') THEN
FNEW = PSAT - PFUNS4(TSAT ,FCO2,ALPHAO2,TMAXO2)
FNEWB = PSAT - PFUNS4(TSATB,FCO2,ALPHAO2,TMAXO2)
ENDIF
C
IF(GASNAME.EQ.'N2') THEN
FNEW = PSAT - PFUNS4(TSAT ,FCN2,ALPHAN2,TMAXN2)
FNEWB = PSAT - PFUNS4(TSATB,FCN2,ALPHAN2,TMAXN2)
ENDIF
DFDT = (FNEWB-FNEW)/(TSATB-TSAT)
IF(DFDT.EQ.0.0) THEN
WRITE(6,*) 'SATNC HAD ZERO DERATIVE'
STOP
ENDIF
DELT = - FNEW/DFDT
SIGNO = SIGNN
SIGNN = SIGN(1.,DELT)
IF(NLOOP.GT.2 .AND. SIGNO.NE.SIGNN) EXTRP = 0.5*EXTRP
IF(ABS(DELT).GT.EXTRP*TSAT) DELT = EXTRP*TSAT*SIGN(1.,DELT)
TSAT = TSAT + DELT
IF(ABS(DELT/TSAT).LT.1.E-6) GO TO 20
10 CONTINUE
WRITE(6,*) 'SATNC DID NOT CONVERGE ON TSAT'
STOP
20 CONTINUE
ENDIF
C
RETURN
END
SUBROUTINE RHOVNC(GASNAME,TEMP,PRES,RHOV)
C
C       IDEAL GAS LAW ASSUMED FOR NON-CONDENSIBLE VAPOR DENSITIES
C
REAL*4 MWN2, MWO2
CHARACTER*3 GASNAME
C
DATA MWO2 / 31.9994 /
DATA MWN2 / 28.0134 /
DATA RGCON / 8314.34 /
C
IF(GASNAME.EQ.'O2') THEN
RHOV = MWO2*PRES/(RGCON*TEMP)
ENDIF
C
IF(GASNAME.EQ.'N2') THEN
RHOV = MWN2*PRES/(RGCON*TEMP)
ENDIF
C
RETURN
END
FUNCTION PFUND2(TEMP,DC,TCRIT)
C
DIMENSION DC(*)
REAL*8 DC
C
RHO = 0.
CAPX = 1.-TEMP/TCRIT
C
DO I= 1,6
RHO = RHO + DC(I)*CAPX**(FLOAT(I-1)/3.)
ENDDO
C
PFUND2 = RHO
FUNCTION PFUNS4(TEMP, FC, ALPHA, TCRIT)

DIMENSION FC(*)
REAL*8 FC

XLNP = FC(1)/TEMP + FC(2) + FC(3)*TEMP
. + FC(4)*MAX(0., TCRIT-TEMP)**ALPHA
. + FC(5)*TEMP**3 + FC(6)*TEMP**4 + FC(7)*TEMP**5
. + FC(8)*TEMP**6 + FC(9)*ALOG(TEMP)

PFUNS4 = EXP(XLNP)

RETURN
END