INL/EXT-07-12493
Rev. 1

# Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

Brad J. Merrill

May 2007


The INL is a U.S. Department of Energy National Laboratory
operated by Battelle Energy Alliance

INL/EXT-07-12493
Rev. 1

# Recent Updates to the MELCOR 1.8.2 Code for ITER Applications

**Brad J. Merrill**

**May 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 Date |
| Lee C. Cadwallader/Technical Checker | 4-6-2007 Date |
| Richard L. Moore/Peer Reviewer | 4/10/07 Date |
| Phil Sharpe/Approver/Manager | 4/6/07 Date |

# 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



# FIGURES



# TABLES

# 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.[4] The new version of MELCOR 1.8.2 was used for the ITER-FEAT Generic Site Safety Report (GSSR)[5] 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 ($T_2$).[6] 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:

$$\frac{\partial M^{a}_{i_{HTO}}}{\partial t} = \dot{M}^{b}_{i_{H_2O}} x^{p}_{i_{HTO}} - \dot{M}^{c}_{i_{H_2O}} x^{a}_{i_{HTO}} + \sum_{j} A_{c_j} v^{a}_{j} M^{a}_{j_{HTO}} / V^{a}_{j} - \sum_{i} A_{c_i} v^{a}_{i} M^{a}_{i_{HTO}} / V^{a}_{i} - \sum_{l} A_{s_l} \Gamma^{c}_{l_{H_2O}} x^{a}_{i_{HTO}} \quad (1)$$

for the atmosphere of the "ith" control volume, where

$M^{a}_{i_{HTO}}$ is the mass of HTO in the atmosphere (kg)

$\dot{M}^{b}_{i_{H_2O}}$ is the bulk pool $H_2O$ boiling rate (kg/s)

$x^{p}_{i_{HTO}}$ is the HTO to $H_2O$ mass ratio in the pool

$\dot{M}^{c}_{i_{H_2O}}$ is the bulk atmosphere $H_2O$ condensation rate (kg/s)

$x^{a}_{i_{HTO}}$ is the HTO to $H_2O$ mass ratio in the atmosphere

$A_{c_{i,j}}$ 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 ($m^3$)

$A_{s_l}$ is the surface area of the "lth" heat structure ($m^2$)

$\Gamma^c_{l_{H_2O}}$ is the $H_2O$ condensation flux on the "lth" heat structure ($kg/m^2$-s)

and, as:

$$\frac{\partial M^p_{i_{HTO}}}{\partial t} = -\overset{\bullet}{M}{}^b_{i_{H_2O}} x^p_{i_{HTO}} + \overset{\bullet}{M}{}^c_{i_{H_2O}} x^a_{i_{HTO}} + \sum_j A_{c_j} v^p_j M^p_{j_{HTO}} / V^p_j - \sum_i A_{c_i} v^p_i M^p_{i_{HTO}} / V^p_i + \sum_l A_{s_l} \Gamma^c_{l_{H_2O}} x^a_{i_{HTO}} \quad (2)$$

for the pool of the "ith" control volume, where

$M^p_{i_{HTO}}$ 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.

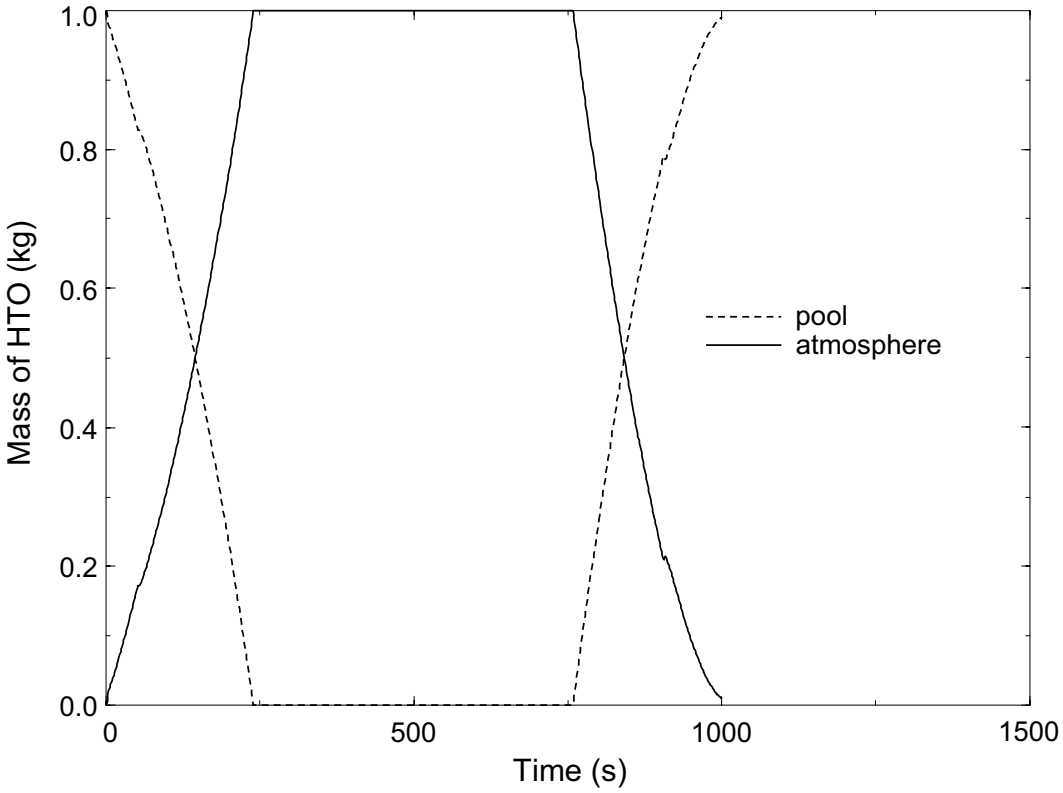


**Figure 1.** HTO pool and atmosphere masses for HTO transport test problem one.

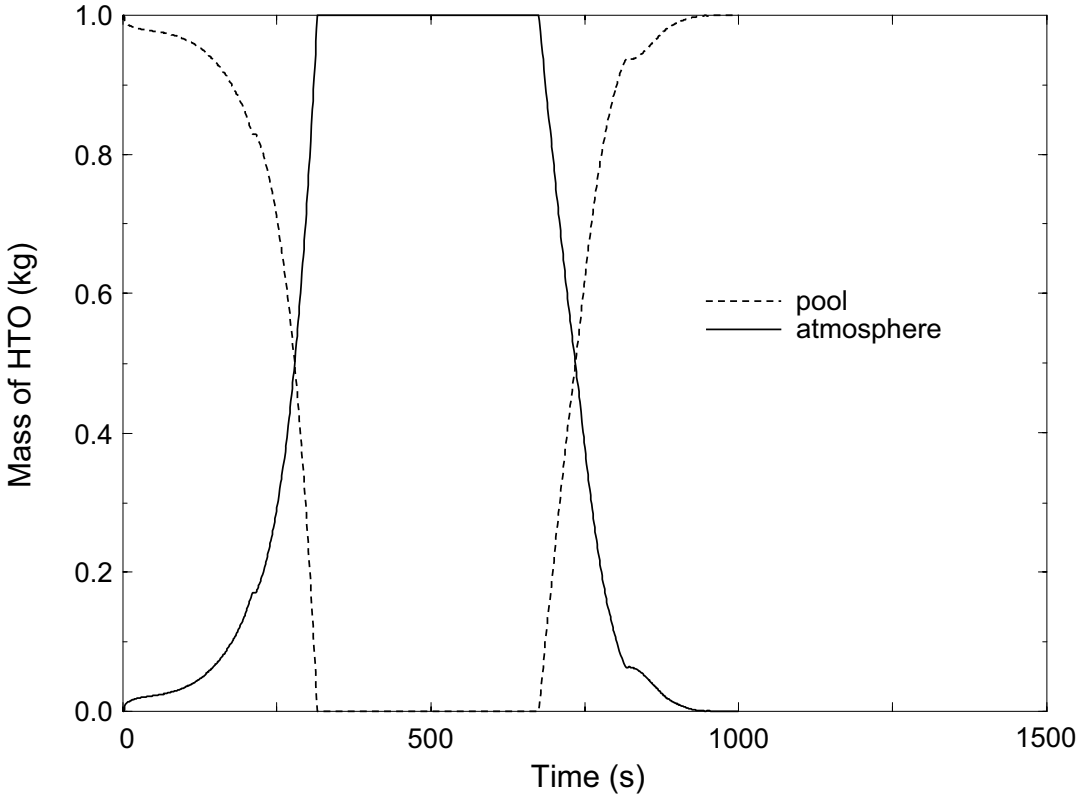


**Figure 2.** HTO pool and atmosphere masses for HTO transport test problem two.

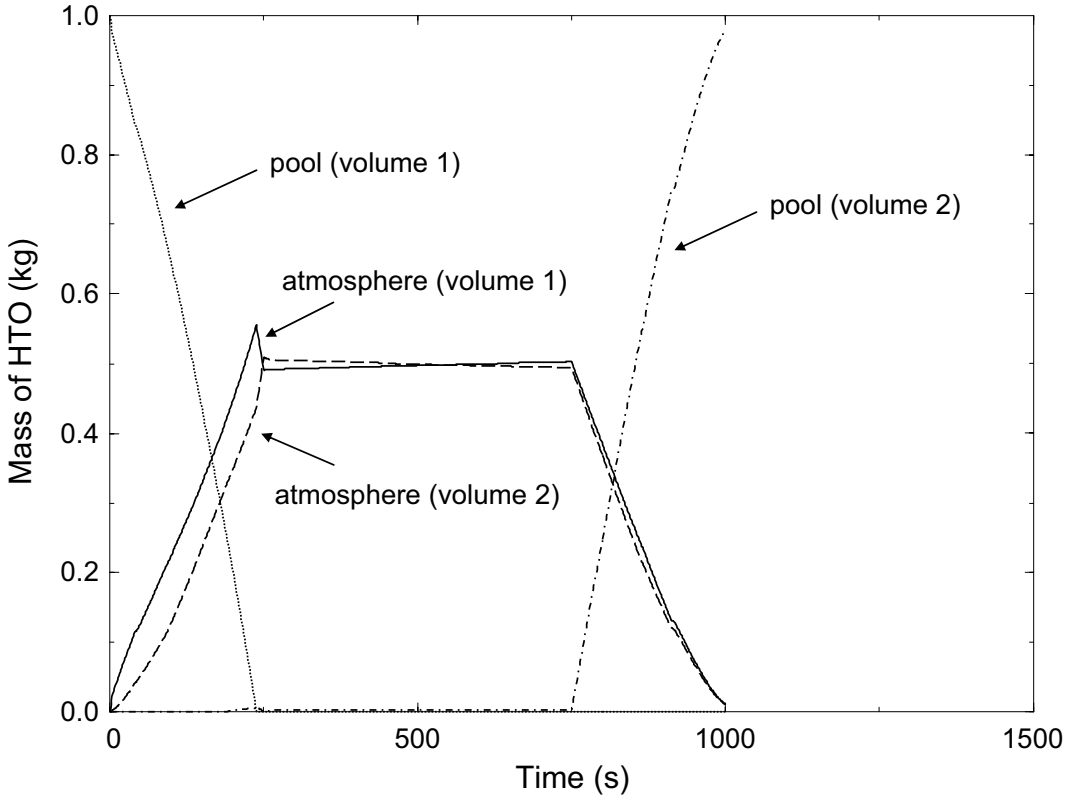


**Figure 3.** HTO pool and atmosphere masses for HTO transport test problem three.

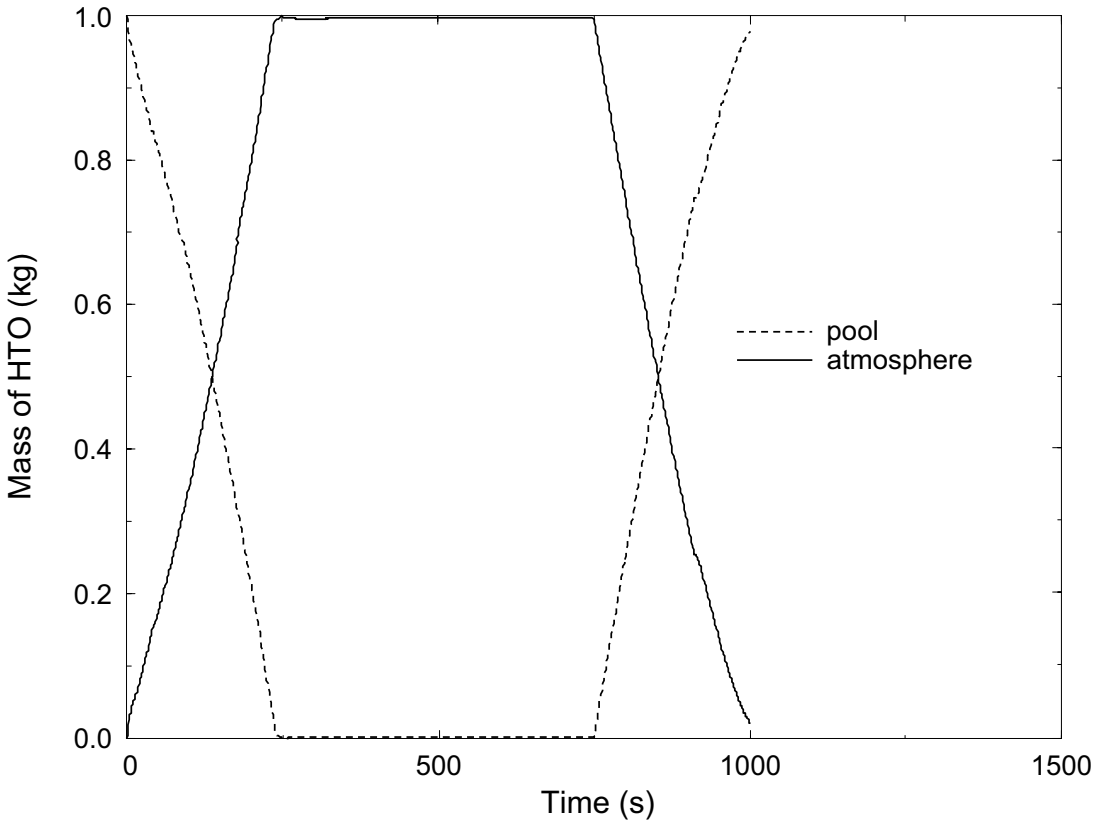


**Figure 4.** Total HTO pool and atmosphere masses for HTO transport test problem three.

# 3. OXIDATION CORRELATION UPDATE

The version of MELCOR 1.8.2 used for the ITER-FEAT GSSR[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.[2] The changes documented in this section update these correlations and add those for air.

In the previous version of this code,[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[3] and converted to kg-Be/m$^2$-s, are:

$$R_{Be-H_2O} = sP_M\left(0.1205 \times e^{-13{,}465/T}\right) \qquad T < 773\ K \qquad (3)$$

$$R_{Be-H_2O} = sP_M\left(4.378 \times 10^{15} \times e^{-42{,}933/T}\right) \qquad 773\ K \le T \le 873\ K \qquad (4)$$

$$R_1 = 49.398 \times e^{-12{,}500/T}$$

$$R_2 = 2.571 \times 10^{7} \times e^{-28{,}789/T} \qquad T < 1133\ K$$

$$R_2 = 35.357 \times e^{-13{,}387/T} \qquad T \ge 1133\ K$$

$$R_{Be-H_2O} = sP_M\sqrt{R_1 R_2} \qquad T > 873\ K \qquad (5)$$

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

$$P_M = (P_{H_2O}/P_o)^{0.9} \quad (6)$$

where $P_{H2O}$ 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 kg-C/$m^2$-s, are:

$$R_{C-H_2O} = sP_M\left(8.571\times10^7 \times e^{-70{,}800/RT}\right) \qquad T < 1470\ K \quad (7)$$

$$R_{C-H_2O} = sP_M\left(5.196 \times e^{-22{,}500/RT}\right) \qquad T \geq 1470\ K \quad (8)$$

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

$$P_M = (P_{H_2O}/P_o) \quad (9)$$

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 kg-W/$m^2$-s, is:

$$R_{W-H_2O} = sP_M\left(41.238 \times e^{-33{,}106/RT}\right) \qquad T < 1470\ K \quad (10)$$

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

$$P_M = (P_{H_2O}/P_o)^{0.78} \quad (11)$$

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 kg-Be/$m^2$-s, are:

$$R_{Be-O_2} = sP_M\left(4.833\times10^5 \times e^{-26{,}200/T}\right) \qquad T < 1073\ K \quad (12)$$

$$R_{Be-O_2} = sP_M\left(34.83 \times e^{-15{,}900/T}\right) \qquad T \geq 1073\ K \quad (13)$$

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

$$P_M = (P_{O_2}/P_o) \quad (14)$$

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 ($P_o = 0.181 \times 10^5$ 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:

$$R_{C-O_2} = sP_M\left(0.2472 \times e^{-5710/T}\right) \qquad T < 1273 \text{ K} \quad (15)$$

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

and for CFC taken from Reference 9 are

$$R_{C-O_2} = sP_M\left(1.4754 \times 10^{7} \times e^{-26,128/T}\right) \qquad T < 983 \text{ K} \quad (17)$$

$$R_{C-O2} = sP_M\left(3.6308 \times 10^{1} \times e^{-13,475/T}\right) \qquad 983 \text{ K} \geq T \geq 1448 \text{ K} \quad (18)$$

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

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

$$P_M = (P_{O_2}/P_o) \quad (20)$$

and where $P_o = 0.181 \times 10^5$ 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:

$$R_{W-O_2} = sP_M\left(1.62 \times 10^{6} \times e^{-24,000/T}\right) \qquad T < 973 \text{ K} \quad (21)$$

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

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

$$P_M = (P_{O_2}/P_o)^{0.5} \quad (23)$$

and where $P_o = 0.181 \times 10^5$ 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,[10] 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:

$$\rho_{nom} = x_{H_2O\ droplets} \times \rho_{H_2O} + \left(1 - x_{H_2O\ droplets}\right) \rho_{nom}^{user\ input} \tag{24}$$

where the mass fraction of water droplets ($x_{H2O}$) in the atmosphere of a given control volume is defined as:

$$x_{H_2O\ droplets} = \frac{\sum_{i=1}^{nbins} m_{14,i}}{\sum_{l=1}^{nclass} \sum_{i=1}^{nbins} m_{l,i}} \tag{25}$$

The nomenclature "nclass" and "nbin" 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.[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:

$$q = \Gamma_c^{O_2} \left( h_v - h_{l_{sat}} \right)_{O_2} + \Gamma_c^{N_2} \left( h_v - h_{l_{sat}} \right)_{N_2} \tag{26}$$

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
and sat = saturated state

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

$$q = \Gamma_c^{O_2} h_{v,l_{O_2}} + \Gamma_c^{N_2} h_{v,l_{N_2}} \tag{27}$$

with

$$h_{v,l} = c_{p_v} \left( T_v - T_l \right) + h_{fg_{sat}} \tag{28}$$

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 Clapeyorn's Equation as follows:

$$h_{fg_{sat}} = T \left( \frac{1}{\rho_v} - \frac{1}{\rho_f} \right) \left. \frac{dp}{dT} \right|_{sat} \tag{29}$$

where

T = saturation temperature/heat structure surface temperature (K)
$\rho_v$ = vapor density (kg/m$^3$)
$\rho_f$ = saturated liquid density (kg/m$^3$)
$\left. \frac{dp}{dT} \right|_{sat}$ = 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:

$$\rho_v = \frac{RT}{p} \tag{29}$$

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:

$$\rho_f = \sum_{i=1}^{6} D_i \left(1 - \frac{T}{T_c}\right)^{(i-1)/3} \tag{30}$$

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

| $T_{liq}$ (K) | $T_v$ (K) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Saturation $h_{l,v}$ (kJ/kg) | | 200 $h_{l,v}$ (kJ/kg) | | 300 $h_{l,v}$ (kJ/kg) | | 400 $h_{l,v}$ (kJ/kg) | |
| | Model | Ref. 7 | Model | Ref. 7 | Model | Ref. 7 | Model | Ref. 7 |
| 60 | 238.26 | 238.26 | 366.63 | 363.12 | 458.32 | 455.76 | 550.01 | 549.21 |
| 70 | 230.50 | 230.50 | 349.70 | 346.47 | 441.39 | 439.11 | 533.08 | 532.56 |
| 80 | 222.30 | 222.30 | 332.33 | 329.79 | 424.02 | 422.43 | 515.71 | 515.88 |
| 90 | 213.23 | 213.23 | 314.09 | 312.99 | 405.78 | 405.63 | 497.45 | 498.78 |

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

| $T_{liq}$ (K) | $T_v$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Saturation $h_{l,v}$ (kJ/kg) | | 200 (K) $h_{l,v}$ (kJ/kg) | | 300 (K) $h_{l,v}$ (kJ/kg) | | 400 (K) $h_{l,v}$ (kJ/kg) | |
| | Model | Ref. 7 | Model | Ref. 7 | Model | Ref. 7 | Model | Ref. 7 |
| 63.15 | 214.83 | 214.83 | 360.19 | 355.05 | 464.02 | 460.25 | 567.85 | 564.89 |
| 70 | 203.14 | 208.14 | 338.12 | 342.04 | 441.95 | 447.24 | 545.78 | 551.88 |
| 80 | 195.65 | 195.65 | 320.25 | 321.55 | 424.08 | 426.75 | 527.91 | 531.39 |
| 90 | 180.64 | 180.64 | 294.85 | 300.62 | 398.68 | 405.82 | 502.51 | 510.39 |

# 6. REFERENCES


1. R. O. Gauntt, et al., MELCOR Computer Code Manuals Vol. 2: Reference Manuals Version 1.8.5, NUREG/CR-6119, Vol.2, Rev. 2, SAND2000-2417/2, Sandia National Laboratories, Albuquerque, NM, May 2000.
2. N. Taylor, "Support and Assistance for MELCOR Quality Assurance and Safety Analysis," ITER Task Agreement ITA 81-18, Task No. C81TD30FU, IDM No ITER-D-259EJK, February, 2007.
3. ITER, "Safety Analysis Data List," G 81 RI 10 03-08-08 W 0.1, Version: 4.0.3 SADL, September 26, (2003)
4. B. J. Merrill, R. L. Moore, S. T. Polkinghorne, D. A. Petti, Modifications to the MELCOR code for application in fusion accident analyses, Fus. Eng. Des. 51-52 (2000) 555-563.
5. "Generic Site Safety Report, Volume VII: Analysis of Reference Events," ITER Report, G 84 RI 6 R0.2, July (2004).
6. N. Taylor, H-W. Bartels, C. Gordon, T. Honda, "Accident Analysis Guidelines 4, Draft Version 4.1.2," ITER Report IDM Number: ITER_D_24TDZ8 v. 1.2, January (2007), p. 13.
7. B. J. Merrill, "Initial Modifications to the MELCOR Code," US ITER Participant Team Report, ITER/US/95/TE/SA-18, June 30 (1995), pp. 22-29.
8. R. O. Gauntt, et al., MELCOR Computer Code Manuals Vol. 2: Reference Manuals Version 1.8.5, NUREG/CR-6119, Vol. 2, Rev. 2, SAND 2000-2417/2, Sandia National Laboratories, Albuquerque, NM, May 2000, pages HS-RM-11-14.
9. Theron Marshall, et al., "SOMBRERO LOVA Analysis Using CFC NB31 Oxidation Data," Fusion Science and Technology, Vol. 45, No. 4, June (2004), p. 592-596.
10. B. J. Merrill, D. L. Hagrman, "MELCOR Aerosol Transport Module Modification for NSSR-1," Idaho National Engineering Laboratory Report, INEL-96/0081, March (1996).
11. B. J. Merrill, "Initial Modifications to the MELCOR Code," US ITER Participant Team Report, ITER/US/95/TE/SA-18, June 30 (1995), pp. 16-21.
12. G. J. Van Wylen, R. E. Sonntag, Fundamentals of Classical Thermodynamics, John Wiley and Sons, Inc., New York, New York, August (1968), p. 600.
13. W. C. Reynolds, Thermodynamic Properties in SI, Department of Mechanical Engineering, Stanford University, Stanford, CA, (1979), p. 119.

# Appendix A

Code Listing of Changes to MELCOR for HTO Transport Model

**CHANGES TO SUBROUTINE CVHRN3**

```
C
******** added after line 458
C
      DIMENSION FTHTX(999), FTTHM(999), FTEXC(999)

C
********* added after line 1139
C
      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
C
*******added after line 1205
C
      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
C
*******added after line 1423
C
      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
C

******* added after line 1728
C
154   DO 155 NVOL=1,NUMVOL
```

```
      DO 155 NMAT=1,3
      ZN(NMAT,NVOL,7) = XMASSN(NMAT,NVOL)
155   CONTINUE
C
******* 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
C
******* added after line 1811
C
        FTTHM(NVOL) = FTTHM(NVOL) + FTEXC(NVOL)
C
******* added after line 1812
C
C + + PRE-CONVECTION TRANSFER
      IF (MONRN2.NE.0) THEN
      CALL RN2BOIL(FTHTX,DTSUB,NUMVOL)
                       ENDIF
C
******* added after line 1825
C
C + + POST-CONVECTION TRANSFER
      IF (MONRN2.NE.0) THEN
      CALL RN2BOIL(FTTHM,DTSUB,NUMVOL)
                       ENDIF

C
```

**END OF CHANGES TO CVHRN3**

CHANGES MADE TO RN1DBC

```
***** 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 RN1DBC

CHANGES MADE TO RN1DBD

```
********** 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
C               COMMON BLOCK WITH DECAY HEAT DATABASE POINTERS
C                  SET UP BY JOHN ORMAN
C                  VERSION 1.06  12/1/86
C
      COMMON /DCHPNT/
C                     ****   CLASS POINTERS   ****
     1                MFPCNM, MFPCID, MCLSMS, MCLSEL, MCLEPT, MCLELM,
     2                MDCHET, MDCTIM, MTIMDH, MTMDHL, MTDALL,
     3                MCLSPB, MCLSPE, MCLSPO, MCLSPN,
     4                NDFPCL, NDELEM, NDDCTM, NDTDAL,
     5                NFPCLS, NELEMS, NDCTIM, NTDALL,
     6                IOSDCH
C
C          ****   CLASS POINTER DESCRIPTIONS   ******
C
C           *** POINTERS TO DATA ARRAYS ***
C
C         MFPCNM - FISSION PRODUCT CLASS NAMES - VARIABLE NAME = FPCNAM
C


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

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

END OF CHANGES TO RN1DBD

**CHANGES TO SUBROUTINE RN1RN4**

********** 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
                      ENDIF
```

*********** added after line3 68**

```
C
C ----- IF HTO TRANPSORT 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)
```

```
  142 CONTINUE
                              ELSE
      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
                            ENDIF
  141 CONTINUE
C
C ----- IF HTO TRANPSORT, INCLUDE CONDENSATION SKIP DEPOSITION
C -----    CONDENSATION ON HS DIRECT TO LIQUID PHASE
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
                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**

```
*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
      DIMENSION FCBOIL(*)
C
*- INCLUDE BLANK
C                                     GENERIC DOUBLE PRECISION VERSION
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, IINTES, INTEGE(NUMINT)
      COMMON /DBLOGC/ NEDLOG, ILOGCS, ILOGES, LOGICA(NUMLOG)
      LOGICAL LOGICA
      COMMON /DBCHAD/ NEDCHR, ICHRCS, ICHRES
      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
      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
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
*-
C
C       CPU TIME AT END
C
        CALL MXXCPU(CPUBEG)
C
        CALL RN2BOL1(FCBOIL, DTSUB, NNVOL,
     +               IRRN1N, XREALX(IRRN1F), IIRN1N, INTEGE(IIRN1F),
     +               ICDCHN, CHARAC(ICDCHF), IIDCHN, INTEGE(IIDCHF) )
C
C       CPU TIME AT END
```

```
C
       CALL MXXCPU(CPUEND)
       CPUR2C=CPUR2C+CPUEND-CPUBEG
C
       RETURN
       END
*DECK RN2BOL1
       SUBROUTINE RN2BOL1(FCBOIL, DTSUB, NNVOL,
     +      NRRN1, XRN1, NIRN1, IRN1, NDCHC, CDCH, IIDCH, IDCH)
C
C
      CHARACTER * 8 CDCH(NDCHC)
      DIMENSION IDCH(IIDCH)
C
*- INCLUDE DCHPNT
C
C               COMMON BLOCK WITH DECAY HEAT DATABASE POINTERS
C                  SET UP BY JOHN ORMAN
C                  VERSION 1.06  12/1/86
C
      COMMON /DCHPNT/
C                     ****   CLASS POINTERS   ****
     1                MFPCNM, MFPCID, MCLSMS, MCLSEL, MCLEPT, MCLELM,
     2                MDCHET, MDCTIM, MTIMDH, MTMDHL, MTDALL,
     3                MCLSPB, MCLSPE, MCLSPO, MCLSPN,
     4                NDFPCL, NDELEM, NDDCTM, NDTDAL,
     5                NFPCLS, NELEMS, NDCTIM, NTDALL,
     6                IOSDCH
C
C          ****   CLASS POINTER DESCRIPTIONS   ******
C
C           *** POINTERS TO DATA ARRAYS ***
C
C         MFPCNM - FISSION PRODUCT CLASS NAMES - VARIABLE NAME = FPCNAM
C
C
C
C       SECOND LEVEL DATA BASE MANAGER ROUTINE
C
C       CALLED BY RN2MOV SUBROUTINE
C
C       INPUT
C         NRRN2 - NUMBER OF REALS IN RN2 DATABASE
C         XRN2  - REAL ARRAY OF RN2 DATABASE VARIABLES
C         NRRN1 - NUMBER OF REALS IN RN1 DATABASE
C         XRN1  - REAL ARRAY OF RN1 DATABASE VARIABLES
C         NIRN1 - NUMBER OF INTEGERS IN RN1 DATABASE
C         IRN1  - INTEGER ARRAY OF RN1 DATABASE VARIABLES
C         NISC  - NUMBER OF INTEGERS IN SCRATCH DATABASE
C         ISC   - INTEGER ARRAY OF SCRATCH DATABASE
C
C       OUTPUT
C         NONE
C
C       EXTERNAL CALLS - RN2MOX
C
C
*- INCLUDE SRM1B
C
C       MELCOR/SR MOD-1
C       MODIFICATION FOR RN FILTER MODEL
C       MELCOR REFERENCE VERSION:   MELCOR V1.80 (SNL)
C       MELCOR/SR CYCLE:            MELCOR/SR MOD-1 CYCLE B
C       UPDATE SEQUENCE IDENTIFIER: SRMOD1B
C
C
*-
      DIMENSION XRN1(NRRN1),IRN1(NIRN1),FCBOIL(*)
C
       SAVE
C
```

```
*- INCLUDE RN2PNT
C
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
C        RN2 DATABASE FOR POOL SCRUBBING AEROSOL REMOVAL
C
        COMMON/RN2PLS/IDBIN,IRRAT,IVRISE,IVSWRM,INUMFP,NUMPLS,NPLS
*-
*- INCLUDE RN2FLT
C
C       RN2 DATABASE FOR FILTER AEROSOL AND FP VAPOR REMOVAL
C
        COMMON/RN2FLT/INMFLT,IDF,IFLTFP,IDFTFA,IDFTFV,
     +  IMXFCA,IMXFTA,IMXFCV,IMXFTV,
     +  IAEFTO,IAEFTN,IRAFTO,IRAFTN,IVPFTO,IVPFTN,IRVFTO,IRVFTN,IOFLT,
     +  NUMFLT,NFLT,ISRFLO,ISRFLN,IOSRFL
*-
*- INCLUDE RN1PNT
C
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,
     +                 IPMCRO, IPMCVO, IPMCRN, IPMCVN, IPEADO, IPEADN,
     +                 IPCLCO, IPRSRA, IPRSRV, IPISRA, IPISRV,
     +                 IPSRCA, IPSRCV, IPIRSP, IPRRSP,
     +                 IPCN1B, IPCN2A, IPCN2B, IPCN3,  IPCN4,  IPCDEP,
     +                 IPCGRW, IPIVDS, IPDSUR, IPFRCR, IPRSMO, IPRSMN,
     +                 IPRCOR, IPRCAV, IPRFDI, IPMRLO, IPMRLN, IPCRCL,
     +                 IPCLVN, IPVNCL, IPMVNO, IPMVNN, IPMFDO, IPMFDN,
     +                 IPMISC, IPNCCB, IPICCB, IPXCCB, IORRN1, IOIRN1,
     +                 NUMSEC, NUMCMP, NUMCLS, NUMSRA, NUMSRV,
     +                 NNSRA,  NNSRV,  NUMCOF, NUMSUR, NNSUR,  IFLAGR,
     +                 NCNODE, NCMATA, NCLC,   NCLCT,  NNCLC,  NNCLCT,
     +                 NVANCL, IPMCRI, ICVPTH,
C  PARAMETERS ADDED BY R.K.COLE, JULY 1991
     +                 NAXLC,  NRADC,  NGAPC,  KCMPC,  NFMATC, IXMFUC,
C  POINTERS ADDED BY R.C.SMITH, ST. PATTY'S DAY 1992
     +                 IPFRLO, IPSDTO, IPFRLN, IPSDTN,
     +                 IPDHAO, IPDHDO, IPDHAN, IPDHDN
*-
*- INCLUDE MXXTIM
C     TIME COMMON BLOCK
C     TIME = PROBLEM TIME
C     DT = TIMESTEP
C     CPU = CPU TIME
C     NCYCLE = CYCLE NUMBER
C     NDTPAK = PACKAGE CONTROLING TIMESTEP
      COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
      COMMON /MXXTPK/ NDTPAK
      CHARACTER*3 NDTPAK
C
*-
C       GET OFFSETS, ETC
C
        IF(MOD(NCYCLE,2).EQ.0) THEN
C
C       EVEN CYCLE
C
               J2=0
```

```
               IF=0
               J1=0
CSR
               IOSR = 0
       ELSE
C
C       ODD CYCLE
C
               J2=IORRN2
               IF=IOFLT
               J1=IORRN1
CSR
               IOSR = IOSRFL
       END IF
C
C       CALL RN2MOX
C
C       FLOW PATH PROCESSES - CONVECTION ONLY
C
       CALL RN2BOL2(FCBOIL, NNVOL,NUMSEC,NUMCLS,DTSUB,
     +  XRN1(IT1AGN-J1),XRN1(IT1ALN-J1),
     +  NDFPCL, NFPCLS, CDCH(MFPCNM), IDCH(MFPCID))
C
       RETURN
       END
*DECK RN2BOL2
       SUBROUTINE RN2BOL2(FCBOIL, NNVOL, NSEC, NCLS, DTSUB,
     .                    AER1G, AER1L,
     +  NDFPCL, NFPCLS, FPCNAM, IDCH )
C
C
C * * HTO MODS TO GET AT CLASS NAMES (BJM 2/20/07)
      CHARACTER * 32 FPCNAM(NDFPCL)
      DIMENSION IDCH(NFPCLS)
C
C
CSR
CSR   INCLUDE 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 CONTROLING TIMESTEP
      COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
      COMMON /MXXTPK/ NDTPAK
      CHARACTER*3 NDTPAK
C
*-
*- 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      KPFSW 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 , KPFSW = 2 , KCVTHR = 3 ,
     1 KCVTYP = 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
```

```
C      KZBJF = ELEVATTION OF BOTTOM OF JUNCTION OPENING IN 'FROM' VOLUME
C      KZTJF = ELEVATTION OF TOP OF JUNCTION OPENING IN 'FROM' VOLUME
C      KZBJT = ELEVATTION OF BOTTOM OF JUNCTION OPENING IN 'TO' VOLUME
C      KZTJT = ELEVATTION OF TOP OF JUNCTION OPENING IN 'TO' VOLUME
C      KL2PF = LENGTH OVER WHICH POOL/ATMOSPHERE FORCE ACTS IN FLOW PATH
C      NFLGEO = DIMENSION OF THE FIRST ARRAY
       PARAMETER ( KFHGTF=1, KFHGTR=2, KZFM=3,   KZTO=4,
     1             KZBJF=5,  KZTJF=6,  KZBJT=7,  KZTJT=8,
     2             KL2PF=9,  NFLGEO=9 )
*-
       DIMENSION AER1G(NSEC,NCLS,NNVOL),AER1L(NCLS,NNVOL),FCBOIL(*)
C
C
C       ADVECTION PORTION OF RUN ROUTINE FOR RN2 PACKAGE
C       CODED BY ROBERT J. GROSS, OCTOBER 1990
C       BASED ON ORIGINAL RN2RN1 CODED BY STEPHEN W. WEBB, JUNE 1986
C
C       CALLED THROUGH RN2MOV AND RNMOW SEQUENCE
C
C
C       INPUT
C         NNVOL  - NUMBER OF CONTROL VOLUMES
C         NSEC   - NUMBER OF SECTIONS
C         NCLS   - NUMBER OF CLASSES
C         AER1G  - AEROSOL MASSES IN GAS PHASE FROM RN1 PACKAGE,
C                  KG, (NSEC,NCLS,NNVOL) VALUES
C         AER1L  - AEROSOL MASSES IN LIQ PHASE FROM RN1 PACKAGE,
C                  KG, (NCLS,NNVOL) VALUES
C
C       OUTPUT
C         AER1G  - AEROSOL MASSES IN GAS PHASE ACCOUNTING
C                  FOR FLOW PATH CONVECTIVE TRANSPORT, KG,
C                  (NSEC,NCLS,NNVOL) VALUES
C         AER1L  - AEROSOL MASSES IN LIQ PHASE ACCOUNTING
C                  FOR FLOW PATH CONVECTIVE TRANSPORT, KG,
C                  (NCLS,NVVOL) VALUES
C                  (NUMFL) VALUES
C
       SAVE
C
*- INCLUDE MXXDTI
C     MXXDTI CONTAINS TIMESTEP INFORMATION
C     DTI (DTOLDI) IS THE TIMESTEP FOR THE PRESENT (LAST) CYCLE
C     DTMAXI (DTMINI) IS THE MAXIMUM (MINIMUM) ALLOWED TIMESTEP
      COMMON /MXXDTI/ DTI , DTOLDI , DTMAXI , DTMINI
*-
*- INCLUDE INOUT
C     COMMON BLOCK CONTAINING INPUT/OUTPUT UNIT NUMBERS
C     NDIAG = DIAGNOSTIC OUTPUT
C     NOUT = STANDARD OUTPUT
C     NPLOTA (NPLOTB) = FIRST (SECOND) PLOT FILE
C     NREST = RESTART FILE
C     NTERMO (NTERMI) = INTERACTIVE TERMINAL OUTPUT (INPUT)
C     NMESAG = SPECIAL MESSAGE FILE
C
      COMMON /INOUT/ NDIAG,NOUT,NPLOTA,NPLOTB,NREST,NTERMO,NTERMI,
     1 NMESAG
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= 3.14159265358979323846264338327 9D0
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
*-
*- INCLUDE RN1SIV
C
C       RN1 SIZE INDEPENDENT VARIABLES
C
        COMMON/RN1SIV/RHONOM,ICOND,ICOEF,VISCOS,DENAIR,FREEMN,
     +  CHI,GAMMA,FSLIP,STICK,PSAT,DELSAT,TURBDS,
     +  TKGOP,DELDIF,DENSTY,FTHERM,VOLUME,
     +  WTCONM,WTMOL,NB2A,NB2B,NB3,NB4,NDEPST,NGROW,
     +  PGAS1,PGAS2,TGAS1,TGAS2,NCLSW,ICRLSE,NCLSBX
*-
*- INCLUDE XMSC
C
C     PARAMETER VALUES FOR XMISC ARRAY
C
C     KSMD = SOURCE GEOMETRIC STANDARD DEVIATION OF PARTICLES
C     KGAS = TOTAL GAS THROUGH MELT IN THE LAST TIME STEP
C     KSTM = MOLE FRACTION OF INCOMING STEAM
C     KWLV = DEPTH OF WATERPOOL USED BY CORCON
C
      PARAMETER (KSSD=1, KGAS=2, KSTM=3, KWLV=4)
C
*-
*- INCLUDE RN2SIV
C
C       RN2 DATABASE SIZE INDEPENDENT VARIABLES
C
        COMMON/RN2SIV/ ICONV,ICOND2
*-
*- INCLUDE MXXEJ6
C     DEBUGGING SWITCH = 0 NO DEBUGGING EDIT
C                      .NE. 0 THEN DEBUGGING EDIT
      COMMON /MXXEJ6/ IVOMIT
C
*-
C
C
C ----- IF HTO TRANPSORT, SUSPEND HTO BY BOILING
C
      DO J=1,NCLS
        IF(IDCH(J).EQ.NCLS) THEN
        IF(FPCNAM(J)(1:3).EQ.'HTO') THEN
        GO TO 10
                                    ENDIF
                             ENDIF
      ENDDO
C
      RETURN
C
   10 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
      CALL OXIDIZW(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  , SPOW  , MATNMS, NUMNAM,
     5             ENINP )
C
      IF(KO2.GE.0) THEN
C
C * * AIR
      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  , SPOW  , MATNMS, NUMNAM,
     5             ENINP )
                   ENDIF
                            ENDIF
```

*********** New Subroutines**

```
      SUBROUTINE OXIDIZA(  NUMHS , NPTOT , NPMAX , NLOC  , IGEOM ,
     1   HSMULT,   XI    , IMAT  , IBVL  , ASURFL, BNDZL , IBVR  ,
     2             ASURFR, BNDZR , HSL   , HVL   , HSR   , HVR   ,
     3             NUMVOL, NNVOL , NUMMAT, PRPRES, XMASS , CVSPC ,
     4             TEMPN , DELM  , DELE  , SPOW  , MATNMS, NUMNAM,
     5             ENINP )
C
C
C THIS SUBROUTINE OXIDIZES BERYILIUM ON HEAT STRUCTURES
C
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
C              STRUCTURE.
C     NLOC   = ARRAY CONTAINING LOCATION OF TEMPERATURE NODES AND MESH
C              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
C              EACH HEAT STRUCTURE.
C     IMAT   = ARRAY CONTAINING MATERIAL NUMBERS FOR MATERIALS IN MESH
C              INTERVALS OF EACH HEAT STRUCTURE.
C     IBVL   = ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS LEFT
C              BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C     ASURFL = ARRAY CONTAINING AREA OF LEFT BOUNDARY SURFACE OF EACH
C              HEAT STRUCTURE.
C     BNDZL  = ARRAY CONTAINING AXIAL LENGTH OF LEFT BOUNDARY SURFACE OF
C              EACH HEAT STRUCTURE.
C     IBVR   = ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS RIGHT
C              BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C     ASURFR = ARRAY CONTAINING AREA OF RIGHT BOUNDARY SURFACE OF EACH
C              HEAT STRUCTURE.
C     BNDZR  = ARRAY CONTAINING AXIAL LENGTH OF RIGHT BOUNDARY SURFACE
C              OF EACH HEAT STRUCTURE.
C     HSL    = ARRAY CONTAINING LEFT SURFACE WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HVL    = ARRAY CONTAINING LEFT VOLUME WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HSR    = ARRAY CONTAINING RIGHT SURFACE WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HVR    = ARRAY CONTAINING RIGHT VOLUME WEIGHTS FOR EACH HEAT
```

```
C               STRUCTURE.
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
      SAVE
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 CONTROLING TIMESTEP
      COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
      COMMON /MXXTPK/ NDTPAK
      CHARACTER*3 NDTPAK
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= 3.14159265358979323846264338327 9D0
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
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   TEMPN(NPTOT)
C
      DIMENSION
     1    XIS(10000)
C
      DIMENSION
C
     1   SPOW(2,NUMHS)                 ,   ENINP(2,NUMHS)
C
*- INCLUDE CVHSP
C      THIS PARAMETER STATMENT CONTAINS POINTERS TO THE SPECIFIC
C      THERMODYNAMIC QUANTITIES IN ARRAYS TSPCN AND TSPCO.
C      DIMENSION:  TSPCN(KSP1DM,NUMMAT,NUMVOL)
C
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
       PARAMETER ( KSPRHO = 1 , KSPH = 2 , KSPE = 3 , KSP1DM = 3 )
*-
      DIMENSION
C
     1   CVSPC(KSP1DM,NUMMAT,NNVOL)
C
*-
*- INCLUDE NCGEOS
C
C     NON-CONDENSIBLE GAS EOS COMMON BLOCKS
C
      PARAMETER (IEOS = 27)
C
      COMMON /NCGNAM/ NAMGAS
C
      CHARACTER*8 NAMGAS(IEOS)
C
      COMMON /NCGEOS/ MAXGAS, NMMAT, RNCG(IEOS), WM(IEOS), EMAXIM(IEOS),
     2 KH2OP  , KH2OLA , KH2OVA , KH2    , KO2    , KCO2   , KCO    ,
     3 KN2    , KNO    , KN2O   , KNH3   , KC2H2  , KCH4   , KC2H4  ,
     4 KHE    , KAR    , KD2    ,
```

```
     5 KGASA  , KGASB  , KGASC  , KGASD  , KGASE  , KGASF  , KGASG  ,
     6 KGASH  , KGASI  , KGASJ  ,
C     SCRATCH STORAGE FOR EOS WORK
     7 XNCGSA(IEOS), XNCGSB(IEOS), XNCGSC(IEOS), XNCGSD(IEOS),
     8 XNCGSE(IEOS), XNCGSF(IEOS), XNCGSG(IEOS), XNCGSH(IEOS)
C
      LOGICAL REZONE, FIRST
C
      DATA NCYCLST /-1/
*-
C
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
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
      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
      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)
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(2:8) .NE. BLANK) THEN
        WRITE(PBUFR,*) MATNAM(2:8)
        READ (PBUFR,*) VOID
                                  ENDIF
                     ENDIF
                            ENDIF
                                                      ENDIF
 3010 CONTINUE
C
      DELMAT  = 0.0
        XMO2    = XMASS(KO2,IVL) + DELM(KO2,IVL)
        PO2     = PRPRES(KO2,IVL)
C
      CALL OXIRATA(MATNMS(MNAME), DT, TEMPN(NODE1), XMO2, ASURFL(I)
     .,            PO2, VOID, OXMO2, OXMCO2, QOX, THKMAT, DELMAT
     .,            HSMULT(I))
C
      O2ENER = 0.
      IF(KO2.GT.0) THEN
      DELM(KO2,IVL) = DELM(KO2,IVL) + OXMO2 *HSMULT(I)
      CALL NCGENR ( KO2, TEMPN(NODE1), O2ENER, O2ENTH)
                 ENDIF
      CO2ENER = 0.
      IF(KCO2.GT.0) THEN
      DELM(KCO2,IVL) = DELM(KCO2,IVL) + OXMCO2 *HSMULT(I)
      CALL NCGENR ( KCO2, TEMPN(NODE1), CO2ENER, CO2ENTH)
                 ENDIF
      DELE(2,IVL) = DELE(2,IVL)
     .                 + OXMO2*O2ENER*HSMULT(I)
     .                 + OXMCO2*CO2ENER*HSMULT(I)
C
      SPOW(1,I)  = SPOW(1,I)  + QOX/ASURFL(I)
      ENINP(1,I) = ENINP(1,I) + QOX*DT
C
      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.)
      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
C
      REZONE = .TRUE.
                                     ENDIF
C
C ****** RIGHT SURFACE
C
      MNAME  = 3*(IMAT(MESH2)-1) + 1
        MATNAM = MATNMS(MNAME)
C
      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.              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.'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)
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(2:8) .NE. BLANK) THEN
        WRITE(PBUFR,*) MATNAM(2:8)
        READ (PBUFR,*) VOID
                                   ENDIF
                      ENDIF
                             ENDIF
                                                          ENDIF
 3030 CONTINUE
C
      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))
C
      O2ENER = 0.
      IF(KO2.GT.0) THEN
      DELM(KO2,IVR) = DELM(KO2,IVR) + OXMO2 *HSMULT(I)
      CALL NCGENR ( KO2, TEMPN(NODE2), O2ENER, O2ENTH)
                ENDIF
      CO2ENER = 0.
      IF(KCO2.GT.0) THEN
      DELM(KCO2,IVR) = DELM(KCO2,IVR) + OXMCO2 *HSMULT(I)
      CALL NCGENR ( 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
      NN = 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.'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)-XI(NN-1)-1.E-6)
      DELX  = MAX(DELX , 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
      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 OXIDIZW(  NUMHS , NPTOT , NPMAX , NLOC  , IGEOM ,
     1   HSMULT,   XI    , IMAT  , IBVL  , ASURFL, BNDZL , IBVR  ,
     2             ASURFR, BNDZR , HSL   , HVL   , HSR   , HVR   ,
     3             NUMVOL, NNVOL , NUMMAT, PRPRES, XMASS , CVSPC ,
     4             TEMPN , DELM  , DELE  , SPOW  , MATNMS, NUMNAM,
     5             ENINP )
C
C
C THIS SUBROUTINE OXIDIZES (BE,C,W) HEAT STRUCTURES WITH WATER
C
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
C              STRUCTURE.
C     NLOC   = ARRAY CONTAINING LOCATION OF TEMPERATURE NODES AND MESH
C              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
C              EACH HEAT STRUCTURE.
C     IMAT   = ARRAY CONTAINING MATERIAL NUMBERS FOR MATERIALS IN MESH
C              INTERVALS OF EACH HEAT STRUCTURE.
C     IBVL   = ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS LEFT
C              BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C     ASURFL = ARRAY CONTAINING AREA OF LEFT BOUNDARY SURFACE OF EACH
C              HEAT STRUCTURE.
C     BNDZL  = ARRAY CONTAINING AXIAL LENGTH OF LEFT BOUNDARY SURFACE OF
C              EACH HEAT STRUCTURE.
C     IBVR   = ARRAY CONTAINING INDEX FOR CONTROL VOLUME WHICH IS RIGHT
C              BOUNDARY VOLUME OF EACH HEAT STRUCTURE.
C     ASURFR = ARRAY CONTAINING AREA OF RIGHT BOUNDARY SURFACE OF EACH
C              HEAT STRUCTURE.
C     BNDZR  = ARRAY CONTAINING AXIAL LENGTH OF RIGHT BOUNDARY SURFACE
C              OF EACH HEAT STRUCTURE.
C     HSL    = ARRAY CONTAINING LEFT SURFACE WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HVL    = ARRAY CONTAINING LEFT VOLUME WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HSR    = ARRAY CONTAINING RIGHT SURFACE WEIGHTS FOR EACH HEAT
C              STRUCTURE.
C     HVR    = ARRAY CONTAINING RIGHT VOLUME WEIGHTS FOR EACH HEAT
C              STRUCTURE.
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
      SAVE
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 CONTROLING TIMESTEP
      COMMON /MXXTIM/ TIME, DT, CPU, NCYCLE
      COMMON /MXXTPK/ NDTPAK
      CHARACTER*3 NDTPAK
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= 3.14159265358979323846264338327 9D0
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
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)
```

```
C
      DIMENSION
C
     1   TEMPN(NPTOT)
C
      DIMENSION
     1    XIS(10000)
C
      DIMENSION
C
     1   SPOW(2,NUMHS)                ,   ENINP(2,NUMHS)
C
*- INCLUDE CVHSP
C      THIS PARAMETER STATMENT CONTAINS POINTERS TO THE SPECIFIC
C      THERMODYNAMIC QUANTITIES IN ARRAYS TSPCN AND TSPCO.
C      DIMENSION:  TSPCN(KSP1DM,NUMMAT,NUMVOL)
C
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
       PARAMETER ( KSPRHO = 1 , KSPH = 2 , KSPE = 3 , KSP1DM = 3 )
*-
      DIMENSION
C
     1   CVSPC(KSP1DM,NUMMAT,NNVOL)
C
*-
*- INCLUDE NCGEOS
C
C     NON-CONDENSIBLE GAS EOS COMMON BLOCKS
C
      PARAMETER (IEOS = 27)
C
      COMMON /NCGNAM/ NAMGAS
C
      CHARACTER*8 NAMGAS(IEOS)
C
      COMMON /NCGEOS/ MAXGAS, NMMAT, RNCG(IEOS), WM(IEOS), EMAXIM(IEOS),
     2 KH2OP  , KH2OLA , KH2OVA , KH2    , KO2    , KCO2   , KCO    ,
     3 KN2    , KNO    , KN2O   , KNH3   , KC2H2  , KCH4   , KC2H4  ,
     4 KHE    , KAR    , KD2    ,
     5 KGASA  , KGASB  , KGASC  , KGASD  , KGASE  , KGASF  , KGASG  ,
     6 KGASH  , KGASI  , KGASJ  ,
C     SCRATCH STORAGE FOR EOS WORK
     7 XNCGSA(IEOS), XNCGSB(IEOS), XNCGSC(IEOS), XNCGSD(IEOS),
     8 XNCGSE(IEOS), XNCGSF(IEOS), XNCGSG(IEOS), XNCGSH(IEOS)
C
      LOGICAL REZONE
C
      DATA NCYCLST /-1/
*-
C
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
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
      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
      REZONE = .FALSE.
      MNAME = 3*(IMAT(MESH1)-1) + 1
      IF((MATNMS(MNAME).EQ.'BERYLLIU'
     .   .OR. MATNMS(MNAME).EQ.'CARBON  '
     .   .OR. MATNMS(MNAME).EQ.'CFC     '
     .   .OR.      MATNMS(MNAME).EQ.'TUNGSTEN')
     .   .AND.               IVL.GT.0           ) THEN
C
      THKMAT = 0
      NN = NODE1
      DO 3010 NM = MESH1, MESH2
      NN = NN + 1
      MNAMX = 3*(IMAT(NM)-1) + 1
      IF(MATNMS(MNAMX).EQ.'BERYLLIU'
     .  .OR.  MATNMS(MNAMX).EQ.'CARBON  '
     .  .OR.  MATNMS(MNAME).EQ.'CFC     '
     .  .OR.       MATNMS(MNAMX).EQ.'TUNGSTEN') THEN
      DELX  = XI(NN) - XI(NN-1)
      IF(DELX.GT.1.1E-6) THKMAT = THKMAT + DELX
                                            ENDIF
 3010 CONTINUE
C
      DELMAT  = 0.0
        XMH2O   = XMASS(3,IVL) + DELM(3,IVL)
        PH2O    = PRPRES(3,IVL)
C
      CALL OXIRATW(MATNMS(MNAME), DT, TEMPN(NODE1), XMH2O, ASURFL(I)
     .,            PH2O, OXMH2, OXMH2O, OXMCO, QOX, THKMAT, DELMAT
     .,            HSMULT(I))
C
```

```
      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
C
      DELE(2,IVL) = DELE(2,IVL)
     .                  + CVSPC(KSPE,3,IVL)*OXMH2O*HSMULT(I)
     .                  + OXMH2*H2ENER*HSMULT(I)
     .                  + OXMCO*COENER*HSMULT(I)
C
      SPOW(1,I)  = SPOW(1,I)  + QOX/ASURFL(I)
      ENINP(1,I) = ENINP(1,I) + QOX*DT
C
      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
C
      REZONE = .TRUE.
                                    ENDIF
C
C ****** RIGHT SURFACE
C
      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')
     .   .AND.               IVR.GT.0           ) THEN
C
      THKMAT = 0
      NN = NODE2 + 1
      DO 3030 NM = MESH2, MESH1, -1
      NN = NN - 1
      MNAMX = 3*(IMAT(NM)-1) + 1
      IF(MATNMS(MNAMX).EQ.'BERYLLIU'
     .  .OR.  MATNMS(MNAMX).EQ.'CARBON  '
     .  .OR.  MATNMS(MNAME).EQ.'CFC     '
     .  .OR.       MATNMS(MNAMX).EQ.'TUNGSTEN') THEN
      DELX  = XI(NN) - XI(NN-1)
      IF(DELX.GT.1.1E-6) THKMAT = THKMAT + DELX
                                            ENDIF
 3030 CONTINUE
C
      DELMAT  = 0.0
        XMH2O   = XMASS(3,IVR) + DELM(3,IVR)
        PH2O    = PRPRES(3,IVR)
C
      CALL OXIRATW(MATNMS(MNAME), DT, TEMPN(NODE2), XMH2O, ASURFR(I)
     .,            PH2O, OXMH2, OXMH2O, OXMCO, QOX, THKMAT, DELMAT
     .,            HSMULT(I))
```

```
C
      DELM(3  ,IVR) = DELM(3  ,IVR) + OXMH2O*HSMULT(I)
      H2ENER = 0.
      IF(KH2.GT.0)  THEN
      DELM(KH2,IVR) = DELM(KH2,IVR) + OXMH2 *HSMULT(I)
      CALL NCGENR ( KH2, TEMPN(NODE2), H2ENER, H2ENTH)
                  ENDIF
      COENER = 0.
      IF(KCO.GT.0)  THEN
      DELM(KCO,IVR) = DELM(KCO,IVR) + OXMCO *HSMULT(I)
      CALL NCGENR ( KCO, TEMPN(NODE2), COENER, COENTH)
                  ENDIF
C
      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.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
      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: OXMCO2 - MASS OF CO2 PRODUCED
C                OXMO2  - MASS OF O2 CONSUMED
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
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 /
      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 /
```

```
C
      OXMO2  = 0.
      OXMCO2 = 0.
      QOX    = 0.
      IF(XMO2.EQ.0.) RETURN
C
      IF(MNAME.EQ.'BERYLLIU')  THEN
C
C
C        BERYLLIUM & OXYGEN
C
C          2BE + O2 -> 2BeO
C
C  * * * H = F + T*S (PAGE D-45, CRC, 60TH EDITION)
C        HFBEO  IN J/(KG-BEO)
C        ROX  IN (KG-C)/(M2*S) ;  QOX   IN W/M2
C
C * * * PRESSURE DEPENDENCE
C
      PMO2 = PO2/0.181E5
C
      TOX = TSURF
      TOX = MIN(2000.0, TOX )
C
      ROX = SFBE1*4.833E5*EXP(-26200.0/TOX)
      IF(TOX.GT.1073.0) ROX = SFBE2*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/MBEO)
C
      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*(MBEO/MBE)*ASURF
C
                              ENDIF
C
      IF(MNAME.EQ.'CARBON  ' .OR. MNAME.EQ.'CFC     ' ) THEN
C
C        CARBON & OXYGEN
C
C          C + O2 -> CO2
C
C  * * * H = F + T*S (PAGE D-45, CRC, 60TH EDITION)
C        HFCO2  IN J/(KG-CO2)
C        ROX  IN (KG-C)/(M2*S) ;  QOX   IN W/M2
C
C * * * PRESSURE DEPENDENCE
      PMO2 = PO2/0.181E5
C
      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 = 1.4754E+07*EXP(-2.6128E+04/TOX)
      IF(TOX .GT. 983.0 .AND. TOX .LE. 1448.0) THEN
      ROX = 3.6308E+01*EXP(-1.3475E+04/TOX)
                                              ENDIF
      IF(TOX .GT. 1448.D0) ROX = 1.57D-2*EXP(-2260.0/TOX)
                               ENDIF
C
      ROX = VOID*PMO2*SFCA*ROX
      HFCO2  = (-93690.0-1.63*TOX/2.303
     .              +.07E-3*TOX**2-2.*0.23E5/TOX)*(4187./MCO2)
C
      DELTHK = MAX(0.0, DT*ROX/(VOID*RHOCA))
```

```
      IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
      OXMO2 = -DT*ROX*(MO2/MCA)*HSMULT*ASURF
      IF(ABS(OXMO2).GT.0.5*XMO2) ROX = 0.5*ROX*XMO2/ABS(OXMO2)
      DELTHK = MAX(0.0, DT*ROX/(VOID*RHOCA))
      OXMO2 = -DT*ROX*(MO2/MCA)*ASURF
      OXMCO2=  DT*ROX*(MCO2/MCA)*ASURF
      QOX   = -HFCO2*ROX*(MCO2/MCA)*ASURF
C
                                  ENDIF
C
      IF(MNAME.EQ.'TUNGSTEN') THEN
C
C        W + 3/2 O2 -> WO3
C
C  * * * H = F + T*S (PAGE D-50, CRC, 60TH EDITION)
C        HFWO3 IN J/(KG-WO3)
C        ROX  IN (KG-W)/(M2*S) ;  QOX   IN W/M2
C
C * * * PRESSURE DEPENDENCE
      PMO2 = SQRT(PO2/0.181E5)
C
      TOX = TSURF
      ROX = 1.62E6*EXP(-24000.0/TOX)
      IF(TOX.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)
C
      DELTHK = MAX(0.0, DT*ROX/(VOID*RHOW))
      IF(DELTHK.GT.THICK) ROX = ROX*THICK/DELTHK
      OXMO2 = -DT*ROX*(1.5*MO2/MW)*HSMULT*ASURF
      IF(ABS(OXMO2).GT.0.5*XMO2) ROX = 0.5*ROX*XMO2/ABS(OXMO2)
      DELTHK = MAX(0.0, DT*ROX/(VOID*RHOW))
      OXMO2 = -DT*ROX*(1.5*MO2/MW)*ASURF
      QOX   = -HFWO3*ROX*(MWO3/MW)*ASURF
C
                             ENDIF
C
      RETURN
      END
      SUBROUTINE OXIRATW(MNAME, DT, TSURF, XMH2O, ASURF, PH2O, OXMH2
     .,                  OXMH2O, OXMCO, QOX, THICK, DELTHK, HSMULT)
C
C  ***** THIS SUBROUTINE OXIDIZES BE, C, W IN WATER
C
C        INPUT:  TSURF - SURFACE TEMPERATURE
C                XMH2O - MASS OF H2O IN BOUNDARY VOLUME
C                ASURF - SURFACE AREA
C                THICK - BERYLLIUM THICKNESS
C                PH2O  - PARTIAL PRESSURE OF H2O 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
C         OUTPUT: OXMH2  - MASS OF H2 PRODUCED
C                 OXMCO  - MASS OF CO PRODUCED
C                 OXMH2O - MASS OF H2O CONSUMED
C                 QOX    - HEATFLUX
C                 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 /
```

```
      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 /
C
      OXMH2  = 0.
      OXMH2O = 0.
      OXMCO  = 0.
      QOX    = 0.
      IF(XMH2O.EQ.0.) RETURN
C
      IF(MNAME.EQ.'BERYLLIU')  THEN
C
C
C --- Beryllium/steam oxidation
C     Be + H2O --> BeO + H2
C
C * * Get hydrogen production and reaction energy
C
C
C * * * Beryllium
C
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
      IF(TOX.GT.873.) THEN
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
                  ENDIF
C
      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
      OXMH2O= -DT*ROX*(MH2O/MBE)*HSMULT*ASURF
      IF(ABS(OXMH2O).GT.0.5*XMH2O) ROX = 0.5*ROX*XMH2O/ABS(OXMH2O)
      DELTHK = MAX(0.0, DT*ROX/RHOBE)
      OXMH2O= -DT*ROX*(MH2O/MBE)*ASURF
      OXMH2 =  DT*ROX*(MH2/MBE)*ASURF
      QOX   = -(HFBEO-HFH2O*(MH2O/MBEO))*ROX*(MBEO/MBE)*ASURF
C
                            ENDIF
C
      IF(MNAME.EQ.'CARBON  ' .OR. MNAME.EQ.'CFC     ' ) THEN
C
```

```
C        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
      PMH2O = PH2O/0.86E5
C
      TOX = TSURF
      TOX = MIN(2000.0, TOX )
      ROX = 8.571E7*EXP(-70800./(RCON*TOX))
      IF(TOX.GT.1470.) ROX = 5.196*EXP(-22500./(RCON*TOX))
      ROX = PMH2O*SFCA*ROX
      HFCO  = (-25400.0-2.05*TOX/2.303
     .              -.00027*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
      OXMCO =  DT*ROX*(MCO/MCA)*ASURF
      QOX   = -(HFCO-HFH2O*MH2O/MCO)*ROX*(MCO/MCA)*ASURF
C
                                  ENDIF
C
      IF(MNAME.EQ.'TUNGSTEN') THEN
C
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
      PMH2O = (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/RHOW)
      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/RHOW)
      OXMH2O= -DT*ROX*(3.*MH2O/MW)*ASURF
      OXMH2 =  DT*ROX*(3.*MH2/MW)*ASURF
      QOX   = -(HFWO3-HFH2O*(3.*MH2O/MWO3))*ROX*(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
      ENDDO
      RHONOM = RHONDP
      IF(.NOT.STDMEL) THEN
      IF(TMASS.GT.0.0) THEN
        XMFRAC = H2OMS/TMASS
        RHONOM = RLIQ*XMFRAC + RHONOM*(1.-XMFRAC)
                      ENDIF
                     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
c    array TDBMAR(9,NVOL) (turbulent bend deposition array)
c    contains the following
c    index 1: flag for valid info:
c                                      -1=invalid entry
c                                       1=valid entry
c    index 2: actual volume id
c    index 3: characteristic dimension (i.e. pipe diameter)
c    index 4: number of bends in the bend model
c    index 5: turning angle of the bends (radians)
c    index 6: cross-sectional area over deposition area (bend deposition)
c    index 7: surface roughness for turbulent depostion model
c    index 8: surface film surface tension
c    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
c
c    array TDBMAR(9,NVOL) (turbulent bend deposition array)
c    contains the following
c    index 1: flag for valid info:
c                                     -1=invalid entry
c                                      1=valid entry
c    index 2: actual volume id
c    index 3: characteristic dimension (i.e. pipe diameter)
c    index 4: number of bends in the bend model
c    index 5: turning angle of the bends (radians)
c    index 6: cross-sectional area over deposition area (bend deposition)
c    index 7: surface roughness for turbulent depostion model
c    index 8: surface film surface tension
c    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(ITERM+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(ITERM+1)
                ELSE
      DIFFDEP = FDIFF(TVOL(2,N),RHOBLK,VISGAS,WMGAS,DPART)
                ENDIF
      DIFFDEP = DIFFDEP*(1.-FLAGGV)
C
```

# Appendix D

Code Listing for Air Condensation Model Modifications

********** CHANGES MADE TO SUBROUTINE HSRUN2**

********* added after line 664**

```
C
      PARAMETER (TMIN   = 273.16000D0, TMAX   = 647.24498D0)
*-
*- INCLUDE NCGEOS
C
C     NON-CONDENSIBLE GAS EOS COMMON BLOCKS
C
      PARAMETER (IEOS = 27)
C
      COMMON /NCGNAM/ NAMGAS
C
      CHARACTER*8 NAMGAS(IEOS)
C
      COMMON /NCGEOS/ MAXGAS, NMMAT, RNCG(IEOS), WM(IEOS), EMAXIM(IEOS),
     2 KH2OP  , KH2OLA , KH2OVA , KH2    , KO2    , KCO2   , KCO    ,
     3 KN2    , KNO    , KN2O   , KNH3   , KC2H2  , KCH4   , KC2H4  ,
     4 KHE    , KAR    , KD2    ,
     5 KGASA  , KGASB  , KGASC  , KGASD  , KGASE  , KGASF  , KGASG  ,
     6 KGASH  , KGASI  , KGASJ  ,
C     SCRATCH STORAGE FOR EOS WORK
     7 XNCGSA(IEOS), XNCGSB(IEOS), XNCGSC(IEOS), XNCGSD(IEOS),
     8 XNCGSE(IEOS), XNCGSF(IEOS), XNCGSG(IEOS), XNCGSH(IEOS)
C
      DIMENSION KPONT(IEOS)
      EQUIVALENCE (KPONT(1),KH2OP)
CC
      DATA NCYCLST /-1/
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
      COMMON /AIRCON/
     .    XMCLO2(10000), XMCLN2(10000), XMCLO2S(10000), XMCLN2S(10000)
     .,   XMCRO2(10000), XMCRN2(10000), XMCRO2S(10000), XMCRN2S(10000)
     .,   HMCLO2(10000), HMCLN2(10000), HMCRO2(10000) , HMCRN2(10000)
*-
```

******** added after line 675**

```
C
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
      XMCLO2S(N) = XMCLO2(N)
      XMCLN2S(N) = XMCLN2(N)
      XMCRO2S(N) = XMCRO2(N)
      XMCRN2S(N) = XMCRN2(N)
      ENDDO
                        ELSE
      DO N = 1, NUMHS
      XMCLO2(N) = XMCLO2S(N)
      XMCLN2(N) = XMCLN2S(N)
      XMCRO2(N) = XMCRO2S(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
```

******** changes to line 1133**

```
     I                 XMFLR         , HTICER        , VOLSFR        ,
     J                 FLXMRN2       , FLXMRO2       , 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 = FLXMRO2*ASURFR(I)*DT
      IF(CMO2.GT.0.5*XMASS(KO2,IVR)) CMO2 = 0.5*XMASS(KO2,IVR)
      XMCRO2(I) = MAX(0.,XMCRO2(I)+CMO2)
      DELM(KO2,IVR) = DELM(KO2,IVR) - CMO2*HSMULT(I)
      HFRO2 = CVSPC(KSPH,KO2,IVR) - HMCRO2(I)
      IF(CMO2.GT.0.0) THEN
      DELEAR = DELEAR  - CVSPC(KSPH,KO2,IVR)*CMO2*HSMULT(I)
                       ELSE
      DELEAR = DELEAR  - HFRO2*CMO2*HSMULT(I)
                       ENDIF
      HMCRO2(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 (BJM)
C
      COMMON /AIRCON/
     .    XMCLO2(10000), XMCLN2(10000), XMCLO2S(10000), XMCLN2S(10000)
     .,   XMCRO2(10000), XMCRN2(10000), XMCRO2S(10000), XMCRN2S(10000)
     .,   HMCLO2(10000), HMCLN2(10000), HMCRO2(10000) , HMCRN2(10000)
*_
```

*******  **added after line 156**

```
C * * ACCOUNT FOR CHANGE FOR POSSIBLE N2 AND O2 FILMS
     3        + HMCLO2(I) + HMCLN2(I) + HMCRO2(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) THEN
      PO2 = ZERO
```

```
      IF(KO2.GT.0) PO2 = PRPRES(KO2)
      PN2 = ZERO
      IF(KN2.GT.0) PN2 = PRPRES(KN2)
C
      IF((XMCN2.GT.ZERO .OR. PN2.GT.ZERO) .AND. TSURF.LT.126.2) THEN
C * * * CALCULATE BINARY DIFFUSION COEFFICIENT FOR ANY FLUID (BJM 5/20/02)
        ICNCRN = 3
        TAVE = 0.5*(TMTC+TATMS)
        CALL MPEVAL(NUMDIF,XMASS,ICNCRN,KN2,NUMMAT,'CVH',
     1              NUMMAT,PRES,TAVE,DAB,IGERR)
           IF ( IGERR .NE. 0 ) THEN
             IERR = 1
             DAB  = ZERO
             CALL MESERA('*** MP CALL FROM HSTRAN DAB',' ',' ',0)
           END IF
C
      CALL CONDNC('N2 ', PRES, TATMS, TMTC, DAB, PO2, PN2
     .,                  VISC, XNU, RHO, PR, CLN, XMFLXN2, HFG, RHOSL)
C
      CMN2 = XMFLXN2*ASURF*DT
      IF(CMN2.GT.0.5*XMASS(KN2)) CMN2 = 0.5*XMASS(KN2)
      XMCN2X = MAX(0.,XMCN2+CMN2)
      IF(XMCN2X.LE.0.0 .AND. CMN2.LT.0.0) CMN2 =-XMCN2
      QCNC = HFG*CMN2*HSMULT
      IF(TSURF .GT. 63.15) THEN
      FLMTHK = XMCN2/RHOSL/ASURF
      IF(FLMTHK.GT.FILMAX) CMN2 = (FLMAXM-FLMTHK)*RHOSL*ASURF
                           ENDIF
      XMFLXN2 = CMN2/ASURF/DT
      HMCN2   = HFG
                                                              ENDIF
C
      IF((XMCO2.GT.ZERO .OR. PO2.GT.ZERO) .AND. TSURF.LT.154.58) THEN
C * * * CALCULATE BINARY DIFFUSION COEFFICIENT FOR ANY FLUID (BJM 5/20/02)
       ICNCRN = 3
       TAVE = 0.5*(TMTC+TATMS)
       CALL MPEVAL(NUMDIF,XMASS,ICNCRN,KO2,NUMMAT,'CVH',
     1             NUMMAT,PRES,TAVE,DAB,IGERR)
          IF ( IGERR .NE. 0 ) THEN
            IERR = 1
            DAB  = ZERO
            CALL MESERA('*** MP CALL FROM HSTRAN DAB',' ',' ',0)
                             END IF
C
      CALL CONDNC('O2 ', PRES, TATMS, TMTC, DAB, PO2, PN2
     .,                  VISC, XNU, RHO, PR, CLN, XMFLXO2, HFG, RHOSL)
C
      CMO2 = XMFLXO2*ASURF*DT
      IF(CMO2.GT.0.5*XMASS(KO2)) CMO2 = 0.5*XMASS(KO2)
      XMCO2X = MAX(0.,XMCO2+CMO2)
      IF(XMCO2X.LE.0.0 .AND. CMO2.LT.0.0) CMO2 =-XMCO2
      QCNC = QCNC + HFG*CMO2*HSMULT
      IF(TSURF.GT.54.34) THEN
      FLMTHK = XMCO2/RHOSL/ASURF
      IF(FLMTHK.GT.FILMAX) CMO2 = (FLMAXM-FLMTHK)*RHOSL*ASURF
                           ENDIF
      XMFLXO2 = CMO2/ASURF/DT
      HMCO2   = HFG
                                                            ENDIF
      QCNC  = QCNC/VOLSF/DT
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
      SQRMAB = SQRT(2./(PPT001/XMOLW1+PPT001/XMOLW2))
```

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

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

******* added after line 153**

```
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
      COMMON /AIRCON/
     .    XMCLO2(10000), XMCLN2(10000), XMCLO2S(10000), XMCLN2S(10000)
     .,   XMCRO2(10000), XMCRN2(10000), XMCRO2S(10000), XMCRN2S(10000)
     .,   HMCLO2(10000), HMCLN2(10000), HMCRO2(10000) , HMCRN2(10000)
*-
      PARAMETER (NWCON = 12*10000)
```

******* added after line 175**

```
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
        WRITE (NRESX) (XMCLO2(I),I=1,NWCON)
```

******* added after line 373**

```
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
        READ (NRESX, END=192, ERR=192) (XMCLO2(I),I=1,NWCON)
```

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

******* added after line 414**

```
C
C * * COMMON FOR AIR CONDENSATION (BJM)
C
      COMMON /AIRCON/
     .    XMCLO2(10000), XMCLN2(10000), XMCLO2S(10000), XMCLN2S(10000)
     .,   XMCRO2(10000), XMCRN2(10000), XMCRO2S(10000), XMCRN2S(10000)
     .,   HMCLO2(10000), HMCLN2(10000), HMCRO2(10000) , HMCRN2(10000)
*-
C
C     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
C * * COMMON FOR AIR CONDENSATION (BJM)
C
C LEFT CONDENSED O2 MASS
C
          CALL MXXPLK (NUMHS , NAMES(46) , NAMUNT(46), IHSNUM)
C
C RIGHT CONDENSED O2 MASS
C
```

```
          CALL MXXPLK (NUMHS , NAMES(47) , NAMUNT(47), IHSNUM)
C
C LEFT CONDENSED N2 MASS
C
          CALL MXXPLK (NUMHS , NAMES(48) , NAMUNT(48), IHSNUM)
C
C RIGHT CONDENSED N2 MASS
C
          CALL MXXPLK (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 MXXPLW (PAK   , NUMHS , SCDATA)
C
C RIGHT CONDENSED O2 MASS
C
      DO I = 1,NUMHS
          SCDATA(I) = XMCRO2(I) * HSMULT(I)
      ENDDO
C
          CALL MXXPLW (PAK   , NUMHS , SCDATA)
C
C LEFT CONDENSED N2 MASS
C
      DO I = 1,NUMHS
          SCDATA(I) = XMCLN2(I) * HSMULT(I)
      ENDDO
C
          CALL MXXPLW (PAK   , NUMHS , SCDATA)
C
C RIGHT CONDENSED N2 MASS
C
      DO I = 1,NUMHS
          SCDATA(I) = XMCRN2(I) * HSMULT(I)
      ENDDO
C
          CALL MXXPLW (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/)
```

```
C
```

******* added after line 1524**

```
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 1545**

```
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
```

************ NEW SUBROUTINES**

```
      SUBROUTINE CONDNC(GASNAME, PRES, TEMP, TSURF, DAB, PO2, PN2
     .,                 VISC, XNU, RHO, PR, CLN, XMFLXNC, HFG, RHOSL)
C
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
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
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,TAVE,PO2,RHOV)
      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,TAVE,PN2,RHOV)
      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, HFGS,1)
```

```
      HFG = CPG *(TEMP-TSURF) + HFGS
C
      XMNC = DAB*SH/CLN
      DPSURF = MAX(1., PRES-PSURF )
      XMFLXNC = XMNC*RHOV*ALOG(DPSURF/DPATMS)*EXP(-3000./PRES)
C
      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          PROPERITIES 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.
      RETURN
                                        ENDIF
      ENDDO
      HFG  = 0.
                          ENDIF
      RETURN
                           ENDIF
C
      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.
      RETURN
                                        ENDIF
      ENDDO
      HFG  = 0.
                          ENDIF
      RETURN
                           ENDIF
C * * * IMOD=2
                    ELSE
      TSATP = (1.+BUMP)*TSAT
      TSATM = (1.-BUMP)*TSAT
C
      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)
```

```
      RHOSL = PFUND2(TSAT,DCO2,TMAXO2)
                        ENDIF
C
      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)
      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
     .,               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
      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 /
C
      DATA FCN2 /
     .    8.3944094440D+03, -1.8785191705D+03, -7.2822291650D+00
     .,   1.0228509660D-02,  5.5560638250D-04, -5.9445446620D-06
     .,   2.7154339320D-08, -4.8795359040D-11,  5.0953608240D+02 /
      DATA ALPHAN2 / 1.95 /
      DATA TMAXN2 / 126.2 /
      DATA PMAXN2 / 3.4E6 /
      DATA TMINN2 / 63.15 /
      DATA CPGN2  / 1038.3 /
C
      DATA DCO2 /
     .    4.3615175D+02,  7.5897189D+02, -4.2576866D+02
     .,   2.3487106D+03, -3.0474660D+03,  1.4850169D+03 /
C
      DATA DCN2 /
     .    3.1402991D+02,  4.4111015D+02,  9.4622994D+02
     .,  -2.9067111D+03,  4.4785979D+03, -2.2746914D+03 /
C
      DATA TO2I /
     .    54.34,  60.,  65.,  70.,  75.,  80.,  85.,  90., 90.19,  95.
     .,    100., 105., 110., 115., 120., 125., 130., 135.,  140., 145.
     .,    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 HFSO2 /
     .        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
```

```
     .,    226.53/

C
      DATA TN2I /
     .     63.15,  65.,  70.,  75., 77.35, 80., 85., 90., 95., 100.
     .,     105., 110., 115., 120., 126.2/
      DATA HFGN2/
     .    214.83, 213.32, 203.14, 202.13, 199.15, 195.65, 188.58
     .,   180.64, 171.58, 161.19, 149.17, 134.88, 116.98,  92.60
     .,     0.0/
      DATA HFSN2/
     .       0.0,   3.27,  13.01,  23.23,  28.05,  33.50,  43.84
     .,    54.43,  65.39,  76.79,  88.71, 101.40, 115.55, 132.46
     .,   180.78/
C
      END
      SUBROUTINE SATNC(GASNAME,TSAT,PSAT,IMOD)
C
C       THIS SUBROUTINE CALCULATES SATURATION PRESSURE FOR A GIVEN
C          TEMPERATURE (IMOD=1), OR SATURATION TEMPERATURE FOR A
C          GIVEN PRESSURE (IMOD=2).  THE EQUATIONS USED ARE THOSE
C          FOR O2 AND N2 FOUND IN W.C. REYNOLDS, "THERMODYNAMIC
C          PROPERITIES IN SI"
C
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
C
      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
C
      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
C
      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
C
      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
C
```

```
      RETURN
      END
      FUNCTION PFUNS4(TEMP,FC,ALPHA,TCRIT)
C
      DIMENSION FC(*)
      REAL*8 FC
C
      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)
C
      PFUNS4  = EXP(XLNP)
C
      RETURN
      END
```