# Code Level Changes WACCMX 3.5.07 Tag06

**The following is a list of changes to the code from the WACCM 3.5.07 Tag06 model SVN waccm06_cam3_5_07 branch tag to extend the model into the thermosphere/ionosphere for modules with TI changes. After each module name the location of the module on the bluefire is given along with a summary of changes to the 3.5.07 Tag06 code.**

**'chemistry.F90'**(Initialize defaults for chemistry processing)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/chemistry.F90

subroutine chem_register(Register constituents for chemistry processing):

Line 127: Changed variable cnst_fixed_ubc(1) from .true. to .false. which eliminates the constant fixed upper boundary condition for all species

Lines 152,168,192: Changed variable has_fixed_ubc from .true. to .false. which is a flag used later to check for a fixed upper boundary condition for current species.

**'diffusion_solver.F90'**(Calculate vertical diffusion, including thermal diffusion, species diffusion, and eddy diffusion)

ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/cam/diffusion_solver.F90

Line 22: Use subroutine cnst_get_ind and variable cnst_mw(constituent molecular weights) from 'constituents.F90' and avogad from 'physconst.F90'.

Line 75: Add variables ubflx_H(the upper boundary mass flux), alphath, indx_o, indx_O2, and indx_H variables to access those species in physics arrays.

alphath is the thermal diffusion coefficient (e.g. Roble, 1995). It is usually close to 0 except for species whose molecular weight is significantly larger or smaller than the mean molecular weight.

subroutine *init_vdiff*(Initialize and allocate variable for diffusion calculations):

Line 113: Added calls to cnst_get_ind to get array indices for O2, O, and H species, fill alphath array with all zeros except index for H, and calculate ubflx_H(upward mass flux of H at upper boundary).

subroutine *compute_vdiff*(Calculates vertical diffusion variables):

Line 135: Added constituent dependent variables cpairv(specific heat at constant pressure), rairv(gas constant), mbarv(mean mass), kmvis(molecular viscosity), and kmcnd(thermal conductivity) to interface of the subroutine compute_vdiff

Line 169: Declare constituent dependent variables cpairv(specific heat at constant pressure), rairv(gas constant), mbarv(mean mass), kmvis(molecular viscosity), and kmcnd(conductivity)

Line 212: Declare rairvi(constituent dependent gas constant at interface level)

Line 239: Declare mw_fac(constituent dependent molecular weight factor), kvt(molecular conductivity), and ttemp/ttemp0(temporary temperature arrays).

Line 263: Modify calculation of rhoi(density at interfaces) using rairvi(constituent dependent gas constant).

Line 264: Added ifdef for WACCM_TIPHYS

Line 308: Added constituent dependent variables cpairv(specific heat at constant pressure), rairv(gas constant), mbarv(mean mass), kmvis(molecular viscosity), and kmcnd(conductivity) to call of the routine compute_molec_diff

Line 398: Pass in zero(zero flux at upper boundary) in call to subroutine vd_lu_decomp

Line 404: Pass in zero(zero flux at upper boundary) in call to subroutine vd_lu_solve

Line 414: Added ifdef for WACCM_TIPHYS

Line 415: Call to subroutine vd_lu_qtdecomp added and pass in zero(zero flux at upper boundary) to this routine.

Line 419: Fill ttemp0/ttemp, temporary temperature arrays

Line 423: Call to subroutine vd_lu_solve added and pass in zero(zero flux at upper boundary) to this routine.

*vd_lu_qtdecomp* is LU-Decompose solver specifically for solving molecular heat conduction. In standard WACCM, eddy diffusion is applied to dry-static energy. But the molecular thermal conductivity needs to be applied to temperature, and this new subroutine is added to address this issue.

Line 427: Add calculation of dse(dry static energy) using cpairv(constituent dependent specific heat at constant pressure)

Line 465: Do not update O2 and O. This is for solving molecular diffusion of minor species, thus bypass O and O2 (major species). Major species diffusion is calculated separately.

Line 470: Call to subroutine vd_lu_decomp includes 3-D instead of 1-D mw_fac(molecular weight factor) variable. Also, added variables mbarv(constituent dependent mean mass), pmid(midpoint pressures), pint(interface pressures), t(temperature), and alphath in the call to 'vd_lu_decomp'.

Line 483: Update q(index for H species) with ubflx_H(upper boundary flux)

Line 498: Combine kvh(diffusivity for heat) and kvt(molecular conductivity)/cpairv (specific heat at constant pressure) to get the total thermal diffusion, which will be used by gw_intr.

Line 731: Added new subroutine 'vd_lu_qtdecomp'

Line 737: The following comments added to the subroutine:
!+hi-waccm
! This is LU-Decompose solver specifically for solving molecular heat conduction. In standard WACCM, eddy
! diffusion is applied to dry-static energy. In the extended WACCM, molecular thermal conductivity is
! applied to temperature. -Hanli Liu
!-hi-waccm

'**dp_coupling.F90**' (Takes output from dynamics processing and prepares for input to physics processing and the reverse for temperature, tendencies, and constituents)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/dynamics/fv/dp_coupling.F90

All ifdefs removed

Line 24: Use additional variables from constituents(cnst_get_ind,cnst_mw) and shr_const_mod(shr_const_rgas).

subroutine _d_p_coupling_(Converts and copies dynamics structures/variables to physics structures/variables):

Line 168: Declare new species index variables, species data arrays, and mbarvi(constituent dependent mean mass at interface levels), tint(temperature at interface levels), and cpairvi(interface specific heat at constant pressure) variables. Need to comment these declarations.

Line 530: Added calls to cnst_get_ind to get array indices for O(ixo), O2(ixo2), H(ixh), H2(ixh2), and N(ixn) species.

Line 580: Temporary fix for high phys_state%q H and H2 values and changed fix for high phys_state%q H and H2 values to just a print if a value is found.

Line 593: Loops to calculate phys_state structure variables mbarv(constituent dependent mean mass), rairv(constituent dependent gas constant), cpairv(constituent dependent specific heat at constant pressure), and cappav(ratio of rairv/cpairv)

Line 614: Loops to calculate phys_state structure variables kmvis(molecular viscosity) and kmcnd(molecular conductivity).

Line 638: Call geopotential_t with rairv(constituent dependent gas constant)

Line 647: Calculate initial phys_state variable s(dry static energy) using cpairv(constituent dependent specific heat at constant pressure)

subroutine _p_d_coupling_(Converts and copies physics processing structures/variables to dynamics structures/variables. Also performs domain decomposition and distribute the physics variables).

Line 691: Use cnst_get_ind, cnst_mw from constituents module

Line 745: Declare cpair3v(constituent dependent specific heat at constant pressure), rair3v(constituent dependent gas constant), and cappa3v(ratio of rairv/cpairv)

Line 770: Declare new species index variables(ixo,ixo2,ixh,ixn), species data arrays(mmro,mmro2,mmrh,mmrn2), and mwbar(constituent dependent mean mass)

Line 805: Get cpair3v(constituent dependent specific heat at constant pressure),
rair3v(constituent dependent gas constant), and cappa3v(ratio of rairv/cpairv) from dyn_in structure

Line 992: cpair3v(constituent dependent specific heat at constant pressure),
rair3v(constituent dependent gas constant), and cappa3v(ratio of rairv/cpairv) calculated which in
turn will be used by other dynamics modules. Currently these variables are not used by the dynamics
modules.

'**dyn_comp.F90**'

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/dynamics/fv/dyn_comp.F90

Line 47: Added ifdef for WACCM_TIPHYS and declared variables cpair3v(constituent dependent specific heat),
rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v) in
dyn_import_t structure

Line 74: Added ifdef for WACCM_TIPHYS and declared variables cpair3v(constituent dependent specific heat),
rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v) in
dyn_export_t structure

subroutine dyn_create_interface:

Line 467:Added ifdef for WACCM_TIPHYS and allocate cpair3v(constituent dependent specific heat),
rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v)

Line 475: Set cpair3v(constituent dependent specific heat),
rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v) to zero

Line 503: Added ifdef for WACCM_TIPHYS and point to dyn_in structure cpair3v(constituent dependent specific heat),
rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v) variables with
the equivalent dyn_out variables

'**eddy_diff.F90**'

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/cam/eddy_diff.F90

subroutine compute_eddy_diff

Line 353: Use physics_state type from physics_types

Line 358: Declare structure state of type physics_state

Line 621: Added constituent dependent variables cpairv(specific heat at constant pressure),
rairv(gas constant), mbarv(mean mass), kmvis(molecular viscosity), and kmcnd(thermal
conductivity) to call of subroutine compute_vdiff

'**geopotential.F90**' (Compute temperature and geopotential height from dry static energy and
pressure and compute geopotential height from temperature and pressure)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/cam/geopotential.F90
subroutine geopotential_dse(Compute the temperature and geopotential height at midpoints and
interfaces from dry static energy and pressure):

Line 22: Add rairv(constituent dependent gas constant) and
cpairv(constituent dependent specific heat at constant pressure) to interface of subroutine
geopotential_dse.

Line 55: Declare rairv(constituent dependent gas constant)and
pairv(constituent dependent specific heat at constant pressure).

Line 75-76: Declare rvog(constituent dependent gas constant/gravity) and
zvirv(water vapor gas constant/constituent dependent gas constant).

Line 109: Loop to calculate rvog(constituent dependent gas constant/gravity) and
zvirv(water vapor gas constant/constituent dependent gas constant) declared above.

Line 114: Loop to calculate tv(virtual temperature), t(temperature),
zm(geopotential height at mid level), and zi(geopotential height at interfaces)

subroutine geopotential_t(Compute geopotential height from temperature and pressure):

Line 134: Add rairv(constituent dependent gas constant) to interface of subroutine
geopotential_t.

Line 165: Declare rairv(gas constant) as constituent dependent

Line 182: Declare rvog(gas constant/gravity) as constituent dependent

Line 187: Declare zvirv(gas constant/gravity) as constituent dependent

Line 238: Calculate zm(geopotential height at mid level) and zi(geopotential height at interfaces) using constituent dependent rvog(gas constant/gravity). Also, calculate rvirv(rh2o/rairv - 1) for use in calculation of tv(virtual temperature)

'**gw_drag.F90**'(Compute forcing due to breaking of internal gravity waves from convection, orography, frontal systems.)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/waccm/gw_drag.F90

subroutine gw_drag_prof(Calculate profiles of temperature, density, and Brunt-Vaisalla frequency on interface levels):

Line 1093: Declared variable ktop_gw and assigned a value of 3

Line 1119: Use ktop_gw as top level for loop to get stress profiles
ktop_gw controls the top level where beginning apply GW forcing. This level should stop within the domain according to study by Ted Shepherd.

'**inti.F90**' missing

Moved to physpkg.F90 as subroutine physics_init

'**iondrag.F90**'(Calculate ion drag tendency and apply to horizonal velocities(winds) and calculate joule heating tendency and apply to temperature)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/iondrag.F90

subroutine jouleheat_tend(Calculate joule heating tendency and apply to temperature):

Line 983,991: In write statements, use
qjoule/state%cpairv(constituent dependent specific heat at constant pressure) instead of
qjoule/cpair(specific heat at constant pressure)

Line 1003: Replaced cpair(specific heat) with state%cpairv(constituent dependent specific heat)

New module '**majorsp_diffusion.F90**' to calculate molecular diffusion of major species

Line 1: Removed includes of misc.h and params.h

Line 622: Changed AMAX1 intrinsic to MAX intrinsic function

'**mo_mean_mass.F90**'(Calculate mean mass from species mixing ratios)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/mo_mean_mass.F90

subroutine set_mean_mass(Calculate mean mass from combination of N2, O2, O, and H mixing ratios):

Line 41: Set logical parameter fixed_mbar to .false.

'**mo_tgcm_ubc.F90**'(Get input TIME-GCM upper boundary data)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/mo_tgcm_ubc.F90

subroutine set_tgcm_ubc(Get upper boundary H, H2, H2O, and CH4 from TIME-GCM):

Line 606: Replaced mw_dry(mean mass of dry air at top level) with
mbartop(constituent dependent mean mass at top level) in interface of subroutine set_tgcm_ubc

Line 624: Added declaration of mbartop(constituent dependent mean mass at top level)

Line 660: Added calculation of mmr(mass mixing ratio) before call of cnst_get_ind

Line 664: In calculation of mass mixing ratio, replace
mw_dry(mean mass of dry air at top level) with
mbartop(constituent dependent mean mass at top level)

'**mo_waccm_hrates.F90**'(Calculate short wavelength, long wavelength, EUV, and auroral heating rates)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/mo_waccm_hrates.F90

subroutine waccm_hrates(Calculate short wavelength, long wavelength, EUV, and auroral heating rates):

Line 178: Replaced call to set_mean_mass with loop to calculate constituent dependent
mbar(mean wet atmospheric mass)

Line 229: Replaced call to call_cp with loop to load state%cpairv(constituent dependent specific heat)
into cpair(specific heat)

'**molec_diff.F90**'(Compute molecular conductivity and diffusivity for minor species)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/cam/molec_diff.F90

Line 35: Assigned different value to parameter pwr(exponentiation factor)

subroutine init_molec_diff(Initialize time dependent coefficients):

Line 74: Added declaration of variable indx_H(index of H in the species array)

function compute_molec_diff:(Compute constituent independent terms for molecular diffusivity)

Line 120: Added variables cpairv(constituent dependent specific heat for constant pressure),
rairv(constituent dependent gas constant), mbarv(constituent dependent mean mass),
kmvis(molecular viscosity), and kmcnd(molecular conductivity) to call of function compute_molec_diff

Line 138: Declare new variables passed into subroutine,
cpairv(constituent dependent specific heat for constant pressure),
rairv(constituent dependent gas constant), mbarv(constituent dependent mean mass),
kmvis(molecular viscosity), and kmcnd(molecular conductivity)

Line 157: Modified declaration of variable
mw_fac_out(constituent dependent molecular weight factor on interface levels) to include two
additional dimensions, vertical and horizontal

Line 171: Added declaration of variable mbarvi(constituent dependent mean mass at interface levels)

Line 170: Added ubc_mmr(upper boundary mass mixing ratios), rairv(constituent dependent gas constant),
and mbarv(constituent dependent mean mass) to argument list in call to subroutine ubc_get_vals

Line 181: Added loops to calculate mbarvi(constituent dependent mean mass at interface levels) and
mw_fac_out(constituent dependent molecular weight factor on interface levels)

Line 200: Modified calculation of tint(interface temperature) to no longer use
ubc_t(upper boundary temperature)

May want to use ifdef WACCM_TIPHYS at both places. In the latter case, if WACCM_TIPHYS is not set
then use tint (:ncol,ntop_molec) = ubc_t(:ncol) (currently commented out).
Added ifdef WACCM_TIPHYS for both

Line 204: Added ifdef for WACCM_MOZART
Changed WACCM_MOZART to WACCM_TIPHYS

Line 205: Modified calculation of rhoi(density at interfaces), replacing
rair(gas constant for dry air) with rairv(constituent dependent gas constant)

Line 221: Calculation of kvh(diffusivity for heat) includes kmvis(molecular viscosity) and
kmcnd(molecular conductivity) instead of km_fac(molecular viscosity constant) and
tint(interface temperature)

Line 232: Added ifdef for WACCM_MOZART
Changed WACCM_MOZART to WACCM_TIPHYS

Line 233: Calculation of dse_top(dse at top boundary) includes
cpairv(constituent dependent specific heat for constant pressure) instead of cpair(specific heat)

function vd_lu_qdecomp:(Compute constituent dependent terms for molecular diffusivity and update
terms of the diffusion equation matrix)

Line 249: Added mbarv(constituent dependent mean mass), pmid(midpoint pressures),
pint(interface pressures), t(temperature), and alphat(thermal diffusion factor) to arguments in
vd_lu_qdecomp function call

Line 275: Variable mw_facm(constituent dependent molecular weight factor) declared as a 2D array
instead of a scalar

Line 276: Declarations of mbarv(constituent dependent mean mass), pmid(midpoint pressures),
pint(interface pressures), t(temperature), and alphat(thermal diffusion factor) added

Line 304: Added gradm , and gradt declarations

Line 314: Added declaration of mbarvi(constituent dependent mean mass at interface)

Line 327: Assign rghd and gradm a value of zero

Line 333: Changed loop which calculates rghd , gradm , and gradt to loop over all but the
top and bottom point

Line 335: Calculation of mbarvi(constituent dependent mean mass at interface levels) added

Line 336: Calculation of rghd includes mbarvi(constituent dependent mean mass at interface levels)
instead of mw_dry(molecular weight of dry air)

Line 338: Added calculation of gradm and gradt

Line 344: Added loops to calculate rghd , gradm , and gradt

Line 371: Use mw_facm(constituent dependent molecular weight factor) array instead of scalar in
calculation of kmq(molecular diffusivity for constituent)

Line 373: Modified calculation of wrk1(temporary working array)

Line 385: Added calculation of kmq(molecular diffusivity for constituents),
wrk0(temporary working array), wrk(temporary working array), and
kvq(eddy diffusivity for constituents) and modified calculation of
cc(subdiagonal of diffusion matrix)

Lines 399,409,417: Modified calculation of cb(diagonal of diffusion matrix)

Line 422: Added calculation of kvq(eddy diffusivity for constituents)

'**nlte_lw.F90**'(Calculate non-LTE heating rates)

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/waccm/nlte_lw.F90

subroutine nlte_tend:(Calculate non-LTE factors and apply to appropriate constituents)

ifdefs added to this module

Line 265: Calculation of qout(temporary array for outfld calls of QNO and QRLNLTE) includes
state%cpairv(constituent dependent specific heat for constant pressure) instead of
cpair(specific heat).

par_xsum.F90

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/dynamics/fv/par_xsum.F90
/ptmp/joemci/waccm_stacy_cam3_1_9_brnchT_waccm_12/models/atm/cam/src/dynamics/fv/par_xsum.F90
Change in same location - bug fixed

'**phys_adiabatic.F90**' is missing

Moved to physpkg.F90 as subroutine phys_run1_adiabatic_or_ideal

'**phys_idealized.F90**' is missing

Moved to physpkg.F90 as subroutine phys_run1_adiabatic_or_ideal

'**physics_types.F90**'(Contains utilities related to physics tendencies)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/physics/cam/physics_types.F90

Line 95: Declare new variables in the structure,
cpairv(constituent dependent specific heat for constant pressure),
rairv(constituent dependent gas constant), cappav(ratio of rairv/cpairv),
mbarv(constituent dependent mean mass), kmvis(molecular viscosity), and kmcnd(molecular conductivity)

Line 170: Use variable cnst_mw(constituent molecular weights) from constituents module. Use variable
shr_const_rgas(universal gas constant) from shr_const_mod module

Line 187: Declare integer variables(ixo,ixo2,ixh,ixh2,ixn,ixh2o) for constituent indices. Declare
variables(mmro, mmro2, mmrh, mmrn2) to hold constituent data. Declare
mbarvi(constituent dependent mean mass at interface levels), tint(interface temperature), and
cpairvi(interface specific heat at constant pressure)

Line 215: Removed loops to update dry static energy

Line 220: Call cnst_get_ind to get ixh(index of H in data array) and ixh2(index of H2 in data array)

Line 235: Added check for large H and H2 values and replace high values

Line 287: Using O, O2, N and N2 constituents (should include H ), update state structure variables mbarv(constituent dependent mean mass), rairv(constituent dependent gas constant), cpairv(constituent dependent specific heat for constant pressure), and cappav(ratio of rairv/cpairv)

Line 314: Using O, O2, N and N2 constituents (should include H ), update state structure variables kmvis(molecular viscosity) and kmcnd(molecular conductivity)

Line 338: Added update of s(dry static energy)

Line 346: Update tend%dtdt with state%cpairv(constituent dependent specific heat for constant pressure) instead of cpair(specific heat)

Line 357: In call to subroutine geopotential_dse, added state%rairv(constituent dependent gas constant) and state%cpairv(constituent dependent specific heat for constant pressure)

subroutine physics_dme_adjust:(Adjust dry mass energy back to input state )

Line 690: Added state%rairv(constituent dependent specific heat for constant pressure) and state%cpairv(constituent dependent specific heat for constant pressure) to geopotential_dse interface

subroutine physics_state_copy:(Copy variables in the physics 'state' structure)

Line 755: Loops to fill cpairv(constituent dependent specific heat for constant pressure), rairv(constituent dependent gas constant), mbarv(constituent dependent mean mass), cappav(ratio of rairv/cpairv), kmvis(molecular viscosity), and kmcnd(molecular conductivity) variables in state_out structure

'**physpkg.F90**'(Contains code from previous phys_idealized.F90 and phys_adiabatic.F90)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/physics/cam/physpkg.F90

Line 322: Added use of mspd_inti routine from majorsp_diffusion module

Line 416: Added call to mspd_inti routine

Line 627: Call geopotential_t now with rairv(constituent dependent gas constant)

'**prognostics.F90**'

Missing

Moved to dyn_comp.F90 (See above)

'**radheat.F90**'(Converts shortwave and longwave heating into net heating. nonLTE and eUV)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/physics/waccm/radheat.F90

subroutine radheat_tend:(Compute net shortwave and longwave radiative heating and boundary flux. nonLTE and eUV )

Line 233: Add state%cpairv(constituent dependent specific heat) to arguments in call to subroutine merge_qrs

Line 234: Replace cpair(specific heat) with state%cpairv(constituent dependent specific heat) in calculation of qout(temporary array for outfld call QRS_TOT)

Line 245: Replace cpair(specific heat) with state%cpairv(constituent dependent specific heat) in calculation of qout(temporary array for outfld call QRL_TOT)

subroutine merge_qrs:(Merge shortwave heating rates)

Line 265: Add state%cpairv(constituent dependent specific heat) to arguments of interface to subroutine merge_qrs

Line 279: Add cpairv(constituent dependent specific heat) declaration

Line 289: Replace cpair(specific heat) with cpairv(constituent dependent specific heat) in calculation of hmrg(merged heating rates) for mesophere/lower thermosphere?

Line 296: Replace cpair(specific heat) with cpairv(constituent dependent specific heat) in calculation of hmrg(merged heating rates) for region below mesosphere/lower thermosphere but above original

'**stepon.F90**'(Does looping over time steps and accesses routines to perform physics and dynamics calculations)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/dynamics/fv/stepon.F90

Line 381: Added ifdef for WACCM_TIPHYS and loops to update variables
cpair3v(constituent dependent specific heat), rair3v(constituent dependent gas constant), and cappa3v(constituent dependent rairv3/cpari3v)

'**te_map.F90**'(Map vertical Lagrangian coordinates to normal grid ) .../dynamics/fv/

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/dynamics/fv/te_map.F90

Line 599: Set omga(vertical pressure velocity ) to zero instead of calculating for first level index. (This is currently commented out) Also, added ifdef WACCM_TIPHYS

'**tphysac.F90**'(Compute tendencies related to physics after coupling to land, sea, and ice models including vertical diffusion, planetary boundary layer, and gravity waves)

/ptmp/joemci/waccm06_cam3_5_07/models/atm/cam/src/physics/cam/tphysac.F90

Line 29: Use subroutine mspd_intr from major_diffusion module
Added ifdef WACCM_TIPHYS for use of mspd_intr

Line 200: Added call to subroutine mspd_intr
Added ifdef WACCM_TIPHYS for call to mspd_intr

'**upper_bc.F90**'

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/chemistry/waccm_mozart/upper_bc.F90

subroutine ubc_init:(Initialization of upper boundary condition time independent fields)

Line 122,129,135: Commented out write statements

subroutine ubc_get_vals:(Upper boundary condition interface routine for vertical diffusion and planetary boundary layer )

Line 161: Variables rairv(constituent dependent gas constant) and
mbarv(constituent dependent mean mass) added to interface of subroutine ubc_get_vals

Line 178: Declarations of rairv(constituent dependent gas constant) and
mbarv(constituent dependent mean mass) added

Line 203: Replaced rair(gas constant) with rairv(constituent dependent gas constant) at top interface in calculation of rho_top(density at top interface)

Line 213: Replaced mwdry(mean mass of dry air) with mbarv(constituent dependent mean mass) in call to set_tgcm_ubc

'**vertical_diffusion.F90**'

/ptmp/joemci/waccm04_cam3_5_07/models/atm/cam/src/physics/cam/vertical_diffusion.F90

subroutine vertical_diffusion_tend:(Interface routine for vertical diffusion)

Line 345 and 364: Add state structure variables
cpairv(constituent dependent specific heat for constant pressure),
rairv(constituent dependent gas constant), mbarv(constituent dependent mean mass),
kmvis(molecular viscosity), and kmcnd(molecular conductivity) to arguments in call of subroutine compute_vdiff