In-code documentation for CVMix

Many contributors from GFDL, LANL, and NCAR
GFDL, LANL, and NCAR

August 14, 2012

Contents

1 Routine/Function Prologues

1.0.1 vmix_driver

The stand-alone driver for the CVMix package. The type of mixing comes from the cvmix_nml namelist, and then other namelists contain parameters for the specific mixing methods.

INTERFACE:

```
Program vmix_driver
```

USES:

```
use vmix_kinds_and_types
use vmix_background
use vmix_convection
use vmix_put_get
```

1.1 Fortran: Module Interface vmix_kinds_and_types

This module contains the declarations for all required vertical mixing data types. It also contains several global parameters used by the vmix package, such as kind numbers and string lengths.

REVISION HISTORY:

```
SVN:$Id: vmix_kinds_and_types.F90 39453 2012-08-14 20:01:16Z mlevy@ucar.edu $
SVN:$URL: https://svn-ccsm-models.cgd.ucar.edu/pop2/branches/vmix_project/source/vmix/vmix
```

USES:

```
uses no other modules
```

DEFINED PARAMETERS:

```
! Kind Types:
! The vmix package uses double precision for floating point computations.
integer, parameter, public :: vmix_r8     = selected_real_kind(13), &
                              vmix_strlen = 256

! Global parameters:
! The value for pi is needed for Bryan-Lewis mixing.
real(kind=vmix_r8), parameter, public :: vmix_PI = &
                    2.0_vmix_r8*acos(0.0_vmix_r8)
```

PUBLIC TYPES:

```
! vmix_input_type contains every possible necessary input field for all
! supported types of mixing.
type, public :: vmix_data_type
    integer :: nlev = -1 ! Number of levels in column
                         ! Setting default to -1 might be F95...

    ! Values on interfaces
    real(vmix_r8), allocatable :: visc_iface(:)   ! nlev+1
    real(vmix_r8), allocatable :: diff_iface(:,:) ! nlev+1, 2
    real(vmix_r8), allocatable :: z_iface(:)      ! nlev+1
    real(vmix_r8), allocatable :: dw_iface(:)     ! nlev+1

    ! Values at tracer points
    real(vmix_r8), allocatable :: dens(:)     ! nlev
    real(vmix_r8), allocatable :: dens_lwr(:) ! nlev
    real(vmix_r8), allocatable :: z(:)        ! nlev
    real(vmix_r8), allocatable :: dz(:)       ! nlev
end type vmix_data_type

! vmix_global_params_type contains global parameters used by multiple
! mixing methods.
type, public :: vmix_global_params_type
    integer                       :: max_nlev  ! maximum number of levels
    real(vmix_r8)                 :: prandtl   ! Prandtl number
end type vmix_global_params_type

! vmix_bkgnd_params_type contains the necessary parameters for background
! mixing. Background mixing fields can vary from level to level as well as
! over latitude and longitude.
type, public :: vmix_bkgnd_params_type
    real(vmix_r8), allocatable :: static_visc(:,:) ! ncol, nlev+1
    real(vmix_r8), allocatable :: static_diff(:,:) ! ncol, nlev+1

    ! Note: need to include some logic to avoid excessive memory use
    !       when static_visc and static_diff are constant or 1-D
    logical :: lvary_vertical   ! True => second dim not 1
    logical :: lvary_horizontal ! True => first dim not 1
end type vmix_bkgnd_params_type

! vmix_conv_params_type contains the necessary parameters for convective
! mixing.
type, public :: vmix_conv_params_type
    real(vmix_r8)              :: convect_diff
    real(vmix_r8)              :: convect_visc
end type vmix_conv_params_type
```

1.2 Fortran: Module Interface vmix_put_get

This module contains routines to pack data into the vmix datatypes (allocating memory as necessary) and then unpack the data out. If we switch to pointers, the pack will just point at the right target and the unpack will be un-necessary.

REVISION HISTORY:

```
SVN:$Id: vmix_put_get.F90 39458 2012-08-14 22:13:41Z mlevy@ucar.edu $
SVN:$URL: https://svn-ccsm-models.cgd.ucar.edu/pop2/branches/vmix_project/source/vmix/vmix
```

USES:

```
use vmix_kinds_and_types, only : vmix_r8,                 &
                                 vmix_strlen,             &
                                 vmix_data_type,          &
                                 vmix_global_params_type, &
                                 vmix_bkgnd_params_type
```

PUBLIC MEMBER FUNCTIONS:

```
public :: vmix_put

interface vmix_put
  module procedure vmix_put_int
  module procedure vmix_put_real
  module procedure vmix_put_real_1D
  module procedure vmix_put_bkgnd_real_1D
  module procedure vmix_put_global_params_int
  module procedure vmix_put_global_params_real
end interface vmix_put
```

1.2.1 vmix_put_int

INTERFACE:

```
subroutine vmix_put_int(Vmix_vars, varname, val, opts)
```

DESCRIPTION:

Write an integer value into a vmix_data_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=*),           intent(in) :: varname
integer,                    intent(in) :: val
character(len=*), optional, intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_data_type), intent(inout) :: Vmix_vars
```

1.2.2 vmix_put_real

INTERFACE:

```
subroutine vmix_put_real(Vmix_vars, varname, val, opts)
```

DESCRIPTION:

Write a real value into a vmix_data_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=*),           intent(in) :: varname
real(vmix_r8),              intent(in) :: val
character(len=*), optional, intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_data_type), intent(inout) :: Vmix_vars
```

1.2.3 vmix_put_real_1D

INTERFACE:

```
subroutine vmix_put_real_1D(Vmix_vars, varname, val, opts)
```

DESCRIPTION:

Write an array of real values into a vmix_data_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=*),            intent(in) :: varname
real(vmix_r8), dimension(:), intent(in) :: val
character(len=*), optional,  intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_data_type), intent(inout) :: Vmix_vars
```

1.2.4 vmix_put_bkgnd_real_1D

INTERFACE:

```
subroutine vmix_put_bkgnd_real_1D(varname, Vmix_bkgnd_params, val, &
                                  opts)
```

DESCRIPTION:

Write an array of real values into a vmix_bkgnd_params_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=vmix_strlen),  intent(in)          :: varname
real(vmix_r8), dimension(:), intent(in)          :: val
character(len=vmix_strlen), optional, intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(out) :: Vmix_bkgnd_params
```

1.2.5 vmix_put_global_params_int

INTERFACE:

```
subroutine vmix_put_global_params_int(Vmix_params, varname, val, opts)
```

DESCRIPTION:

Write an integer value into a vmix_global_params_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=*),           intent(in) :: varname
integer,                    intent(in) :: val
character(len=*), optional, intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_global_params_type), intent(inout) :: Vmix_params
```

1.2.6 vmix_put_global_params_real

INTERFACE:

```
subroutine vmix_put_global_params_real(Vmix_params, varname, val, opts)
```

DESCRIPTION:

Write a real value into a vmix_global_params_type variable.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
character(len=*),           intent(in) :: varname
real(vmix_r8),              intent(in) :: val
character(len=*), optional, intent(in) :: opts
```

OUTPUT PARAMETERS:

```
type (vmix_global_params_type), intent(inout) :: Vmix_params
```

1.3 Fortran: Module Interface vmix_background

This module contains routines to initialize the derived types needed for background mixing (either specifying a scalar, 1D, or 2D field for viscosity and diffusivity coefficients or calculating these coefficients using the Bryan-Lewis method) and to set the viscosity and diffusivity coefficients to this specified value.

REVISION HISTORY:

```
SVN:$Id: vmix_background.F90 39453 2012-08-14 20:01:16Z mlevy@ucar.edu $
SVN:$URL: https://svn-ccsm-models.cgd.ucar.edu/pop2/branches/vmix_project/source/vmix/vmix
```

USES:

```
use vmix_kinds_and_types, only : vmix_PI,                 &
                                 vmix_r8,                 &
                                 vmix_data_type,          &
                                 vmix_global_params_type, &
                                 vmix_bkgnd_params_type
```

PUBLIC MEMBER FUNCTIONS:

```
public :: vmix_init_bkgnd
public :: vmix_coeffs_bkgnd

interface vmix_init_bkgnd
  module procedure vmix_init_bkgnd_scalar
  module procedure vmix_init_bkgnd_1D
  module procedure vmix_init_bkgnd_2D
  module procedure vmix_init_bkgnd_BryanLewis
end interface vmix_init_bkgnd
```

1.3.1 vmix_init_bkgnd_scalar

INTERFACE:

```
subroutine vmix_init_bkgnd_scalar(Vmix_bkgnd_params, bkgnd_visc, bkgnd_diff)
```

DESCRIPTION:

Initialization routine for background mixing with a static field. For each column, this routine sets the static viscosity / diffusivity to the given scalar constants.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
real(vmix_r8), intent(in) :: bkgnd_visc
real(vmix_r8), intent(in) :: bkgnd_diff
```

OUTPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(out) :: Vmix_bkgnd_params
```

1.3.2 vmix_init_bkgnd_1D

INTERFACE:

```
subroutine vmix_init_bkgnd_1D(Vmix_params, Vmix_bkgnd_params, &
                              bkgnd_visc, bkgnd_diff, ncol)
```

DESCRIPTION:

Initialization routine for background mixing with a static field. For each column, this routine sets the static viscosity / diffusivity to the given 1D field. If field varies horizontally, need to include ncol!

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
type (vmix_global_params_type), intent(in) :: Vmix_params
real(vmix_r8), dimension(:),    intent(in) :: bkgnd_visc
real(vmix_r8), dimension(:),    intent(in) :: bkgnd_diff
integer, optional,              intent(in) :: ncol
```

OUTPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(out) :: Vmix_bkgnd_params
```

1.3.3 vmix_init_bkgnd_2D

INTERFACE:

```
subroutine vmix_init_bkgnd_2D(Vmix_params, Vmix_bkgnd_params, &
                              bkgnd_visc, bkgnd_diff, ncol)
```

DESCRIPTION:

Initialization routine for background mixing with a static field. For each column, this routine sets the static viscosity / diffusivity to the given 2D field.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
type (vmix_global_params_type), intent(in) :: Vmix_params
real(vmix_r8), dimension(:,:),  intent(in) :: bkgnd_visc
real(vmix_r8), dimension(:,:),  intent(in) :: bkgnd_diff
integer,                        intent(in) :: ncol
```

OUTPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(out) :: Vmix_bkgnd_params
```

1.3.4 vmix_init_bkgnd_BryanLewis

INTERFACE:

```
subroutine vmix_init_bkgnd_BryanLewis(Vmix_vars, Vmix_params,         &
                                      Vmix_bkgnd_params, colid, ncol, &
                                      bl1, bl2, bl3, bl4)
```

DESCRIPTION:

Initialization routine for background mixing with a Bryan-Lewis mixing. For each column, this routine sets the static viscosity / diffusivity based on the specified parameters. Note that the units of these parameters must be consistent with the units of viscosity and diffusivity – either cgs or mks, but don't mix and match!

The Bryan-Lewis parameterization uses the following:

$$\begin{aligned} \kappa_{BL} &= \mathrm{bl1} + \frac{\mathrm{bl2}}{\pi} \tan^{-1}\Big(\mathrm{bl3}(z - \mathrm{bl4})\Big) \\ \nu_{BL} &= \mathrm{Pr} \cdot \kappa_{BL} \end{aligned}$$

This is all based on the following paper:

A Water Mass Model of the World Ocean
K. Bryan and L. J. Lewis
Journal of Geophysical Research, vol 84 (1979), pages 2503-2517.

In that paper,

$\text{bl1} = 8 \cdot 10^{-5}$ m^2/s

$\text{bl2} = 1.05 \cdot 10^{-4}$ m^2/s

$\text{bl3} = 4.5 \cdot 10^{-3}$ m^{-1}

$\text{bl4} = 2500$ m

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
type (vmix_data_type),          intent(in) :: Vmix_vars   ! Depth, nlev
type (vmix_global_params_type), intent(in) :: Vmix_params ! Prandtl
integer,                        intent(in) :: colid, ncol

! Units are first column if Vmix_data%depth is m, second if cm
real(vmix_r8), intent(in)         :: bl1,     &! m^2/s or cm^2/s
                                     bl2,     &! m^2/s or cm^2/s
                                     bl3,     &! 1/m   or 1/cm
                                     bl4       ! m     or cm
```

OUTPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(out) :: Vmix_bkgnd_params
```

1.3.5 vmix_coeffs_bkgnd

INTERFACE:

```
subroutine vmix_coeffs_bkgnd(Vmix_vars, Vmix_bkgnd_params, colid)
```

DESCRIPTION:

Computes vertical diffusion coefficients for static mixing. This routine simply copies viscosity / diffusivity values from Vmix_bkgnd_params to Vmix_vars.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
type (vmix_bkgnd_params_type), intent(in) :: Vmix_bkgnd_params
! Need to know column for pulling data from static_visc and _diff
integer,                       intent(in) :: colid
```

INPUT/OUTPUT PARAMETERS:

```
type (vmix_data_type), intent(inout) :: Vmix_vars
```

1.4 Fortran: Module Interface vmix_convection

This module contains routines to initialize the derived types needed for convective mixing and to set the viscosity and diffusivity in unstable columns.

REVISION HISTORY:

```
SVN:$Id: vmix_convection.F90 39453 2012-08-14 20:01:16Z mlevy@ucar.edu $
SVN:$URL: https://svn-ccsm-models.cgd.ucar.edu/pop2/branches/vmix_project/source/vmix/vmix
```

USES:

```
use vmix_kinds_and_types, only : vmix_r8,               &
                                 vmix_data_type,        &
                                 vmix_conv_params_type
```

PUBLIC MEMBER FUNCTIONS:

```
public :: vmix_init_conv
public :: vmix_coeffs_conv
```

1.4.1 vmix_init_conv

INTERFACE:

```
subroutine vmix_init_conv(Vmix_conv_params, convect_diff, convect_visc)
```

DESCRIPTION:

Initialization routine for convective mixing.

USES:

```
Only those used by entire module.
```

OUTPUT PARAMETERS:

```
type (vmix_conv_params_type), intent(out) :: Vmix_conv_params
```

INPUT PARAMETERS:

```
real(vmix_r8), intent(in) :: &
   convect_diff,      &! diffusivity to mimic convection
   convect_visc        ! viscosity to mimic convection
```

1.4.2 vmix_coeffs_conv

INTERFACE:

```
subroutine vmix_coeffs_conv(Vmix_vars, Vmix_conv_params)
```

DESCRIPTION:

Computes vertical diffusion coefficients for convective mixing.

USES:

```
Only those used by entire module.
```

INPUT PARAMETERS:

```
type (vmix_conv_params_type), intent(in)  :: Vmix_conv_params
```

INPUT/OUTPUT PARAMETERS:

```
type (vmix_data_type), intent(inout) :: Vmix_vars
```