coupler/coupler_main.F90

```
 call mpp_init()
!these clocks are on the global pelist
 initClock = mpp_clock_id( 'Initialization' )
 mainClock = mpp_clock_id( 'Main loop' )
 termClock = mpp_clock_id( 'Termination' )
 call mpp_clock_begin(initClock)

 call fms_init
 call constants_init

 call coupler_init
 if(do_chksum) call coupler_chksum('coupler_init+', 0)
```

coupler/coupler_main.F90 – coupler_init

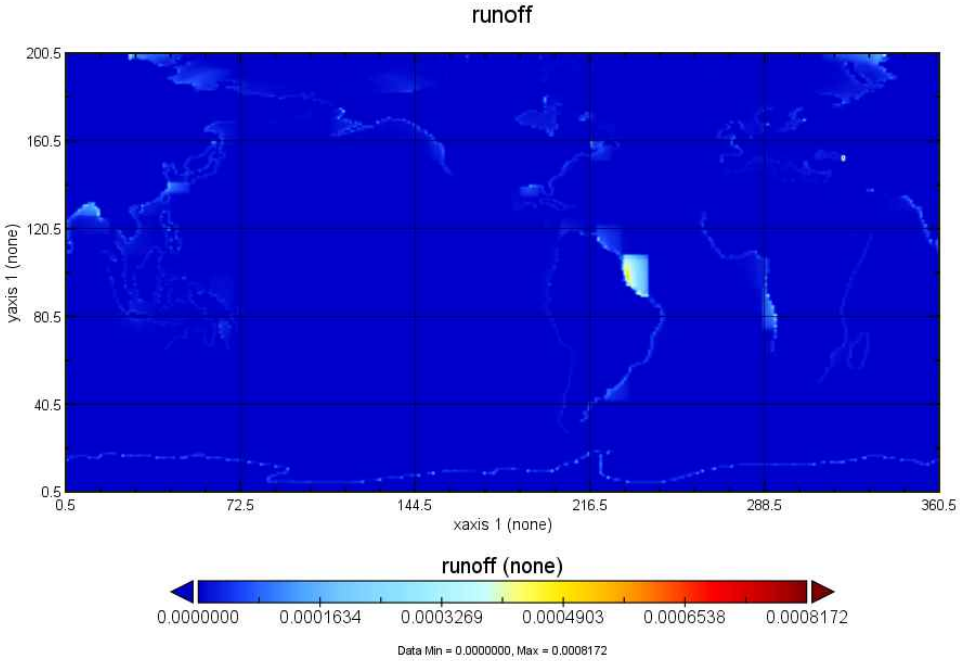
```
- ice -----------
 if( Ice%pe ) then
    call mpp_set_current_pelist(Ice%pelist)
    if( mpp_pe().EQ.mpp_root_pe() ) then
      call DATE_AND_TIME(walldate, walltime, wallzone, wallvalues)
      write(errunit,*) 'Starting to initialize ice model at '&
                       //trim(walldate)//' '//trim(walltime)
    endif
    call ice_model_init( Ice, Time_init, Time, Time_step_atmos, Time_step_cpld )
    if( mpp_pe().EQ.mpp_root_pe() ) then
```

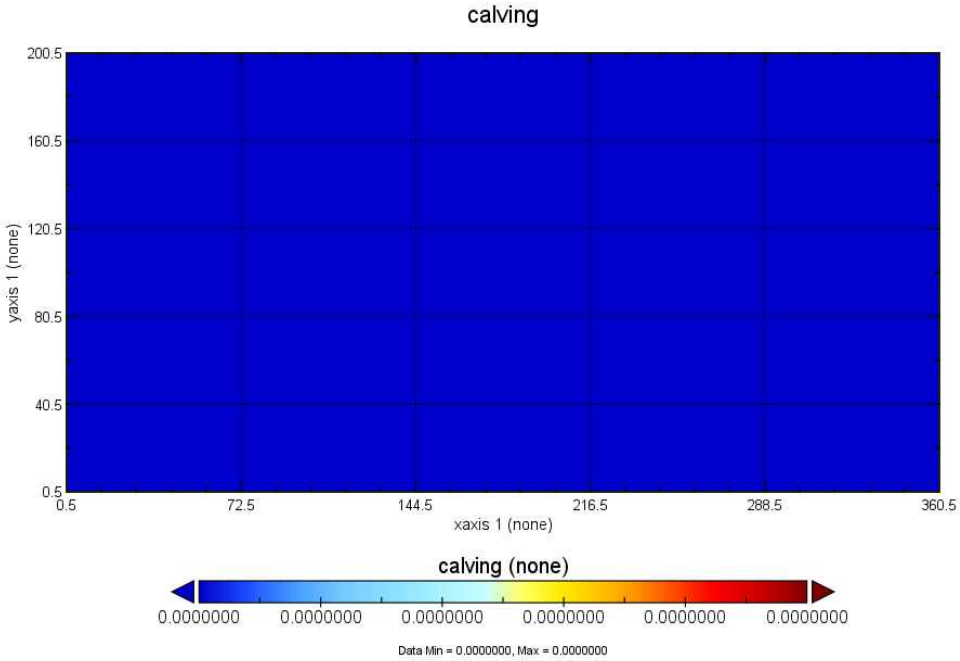
ice_sis/ice_type.F90 – ice_model_init

```
restart_file = 'ice_model.res.nc'
id_restart = register_restart_field(Ice_restart, restart_file, 'part_size', Ice%part_size, domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'albedo',    Ice%albedo,    domain=domain)
id_restart_albedo = register_restart_field(Ice_restart, restart_file, 'albedo_vis_dir', Ice%albedo_vis_dir, &
                                           domain=domain, mandatory=.false.)
id_restart        = register_restart_field(Ice_restart, restart_file, 'albedo_nir_dir', Ice%albedo_nir_dir, &
                                           domain=domain, mandatory=.false.)
id_restart        = register_restart_field(Ice_restart, restart_file, 'albedo_vis_dif', Ice%albedo_vis_dif, &
                                           domain=domain, mandatory=.false.)
id_restart        = register_restart_field(Ice_restart, restart_file, 'albedo_nir_dif', Ice%albedo_nir_dif, &
                                           domain=domain, mandatory=.false.)
id_restart = register_restart_field(Ice_restart, restart_file, 'rough_mom',   Ice%rough_mom,        domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'rough_heat',  Ice%rough_heat,       domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'rough_moist', Ice%rough_moist,      domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 't_surf',      Ice%t_surf,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'h_snow',      Ice%h_snow(:,:,2:km), domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'h_ice',       Ice%h_ice(:,:,2:km),  domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 't_ice1',      Ice%t_ice1(:,:,2:km), domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 't_ice2',      Ice%t_ice2(:,:,2:km), domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'u_ice',       Ice%u_ice,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'v_ice',       Ice%v_ice,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'sig11',       Ice%sig11,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'sig22',       Ice%sig22,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'sig12',       Ice%sig12,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_u',      Ice%flux_u,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_v',      Ice%flux_v,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_t',      Ice%flux_t,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_q',      Ice%flux_q,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_salt',   Ice%flux_salt,        domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'flux_lw',     Ice%flux_lw,          domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'lprec',       Ice%lprec,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'fprec',       Ice%fprec,            domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'runoff',      Ice%runoff,           domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'calving',     Ice%calving,          domain=domain)
id_restart = register_restart_field(Ice_restart, restart_file, 'runoff_hflx', Ice%runoff_hflx,      domain=domain, ma

id_restart = register_restart_field(Ice_restart, restart_file, 'calving_hflx',Ice%calving_hflx,     domain=domain, ma

id_restart = register_restart_field(Ice_restart, restart_file, 'p_surf',      Ice%p_surf,           domain=domain)
id_restart         = register_restart_field(Ice_restart, restart_file, 'flux_sw_vis_dir', Ice%flux_sw_vis_dir, &
                                            domain=domain, mandatory=.false.)
id_restart         = register_restart_field(Ice_restart, restart_file, 'flux_sw_vis_dif', Ice%flux_sw_vis_dif, &
                                            domain=domain, mandatory=.false.)
id_restart_flux_sw = register_restart_field(Ice_restart, restart_file, 'flux_sw_nir_dir', Ice%flux_sw_nir_dir, &
                                            domain=domain, mandatory=.false.)
id_restart         = register_restart_field(Ice_restart, restart_file, 'flux_sw_nir_dif', Ice%flux_sw_nir_dif, &
                                            domain=domain, mandatory=.false.)

     Total SW flux is broken into 4 components in Nalanda,
     it was a single component preNalanda.

restart_file = 'INPUT/ice_model.res.nc'
if (file_exist(restart_file)) then
   call restore_state(Ice_restart)
   if( .NOT. query_initialized(Ice_restart, id_restart_albedo)) then
```

```
0.000000000000000E+000  0.000000000000000E+000  0.000000000000000E+000
0.000000000000000E+000  0.000000000000000E+000  0.000000000000000E+000
0.000000000000000E+000  0.000000000000000E+000  0.000000000000000E+000
0.000000000000000E+000  0.000000000000000E+000  0.000000000000000E+000
0.000000000000000E+000  0.000000000000000E+000  4.531215017777868E-005
4.909091512672603E-005  3.529412424541079E-005  5.460944521473721E-005
2.207468787673861E-005  2.207453690061811E-005  2.207453690061811E-005
2.207453690061811E-005  2.207453508162871E-005  3.427496994845569E-005
4.909091876470484E-005  3.253491013310850E-005  3.253491013310850E-005
3.253491013310850E-005  3.253491013310850E-005  3.253491013310850E-005
5.200459781917743E-005  4.741936572827399E-005  3.115175059065223E-005
3.115175059065223E-005  3.115175059065223E-005  3.115175059065223E-005
3.115175059065223E-005  3.115175059065223E-005  3.115175059065223E-005
3.115175059065223E-005  3.115175059065223E-005  3.115175059065223E-005
3.115175059065223E-005  3.115175059065223E-005  3.115175059065223E-005
3.115175059065223E-005  3.115175059065223E-005  3.115175059065223E-005
3.115175059065223E-005  4.741902375826612E-005  5.200492523727007E-005
3.805342203122564E-005  5.054276334703900E-005  1.103726826841012E-004
0.000000000000000E+000  0.000000000000000E+000  0.000000000000000E+000
```

~/restart/ice_model.res.nc






```
        fprec:units = "none" ;
double runoff(Time, yaxis_1, xaxis_1) ;
        runoff:long_name = "runoff" ;
        runoff:units = "none" ;
double calving(Time, yaxis_1, xaxis_1) ;
        calving:long_name = "calving" ;
        calving:units = "none" ;
double runoff_hflx(Time, yaxis_1, xaxis_1) ;
        runoff_hflx:long_name = "runoff_hflx" ;
        runoff_hflx:units = "none" ;
double calving_hflx(Time, yaxis_1, xaxis_1) ;
        calving_hflx:long_name = "calving_hflx" ;
        calving_hflx:units = "none" ;
```

coupler/coupler_main.F90

```
! Update Ice_ocean_boundary; first iteration is supplied by restart
if( use_lag_fluxes )then
   call mpp_clock_begin(newClock3)
   call flux_ice_to_ocean( Time, Ice, Ocean, Ice_ocean_boundary )
   call mpp_clock_end(newClock3)
end if
```

```
if( Ocean%is_ocean_pe )then
   call mpp_set_current_pelist(Ocean%pelist)
   call mpp_clock_begin(newClock12)

   if (do_chksum) call ocean_chksum('update_ocean_model-', nc)
   ! update_ocean_model since fluxes don't change here

   if (do_ocean) &
     call update_ocean_model( Ice_ocean_boundary, Ocean_state,  Ocean, &
                              Time_ocean, Time_step_cpld )
```

mom5/ocean_core/ocean_model.F90

```
! obtain surface boundary fluxes from coupler
call mpp_clock_begin(id_sbc)
call get_ocean_sbc(Time, Ice_ocean_boundary, Thickness, Dens, Ext_mode,       &
     T_prog(1:num_prog_tracers), Velocity, pme, melt, river, runoff, calving, &
     upme, uriver, swflx, swflx_vis, patm)
call mpp_clock_end(id_sbc)
```

mom5/ocean_core/ocean_sbc.F90

```
subroutine get_ocean_sbc(Time, Ice_ocean_boundary, Thickness, Dens, Ext_mode, T_prog, Velocity, &
                         pme, melt, river, runoff, calving, upme, uriver, swflx, swflx_vis, patm)

  type(ocean_time_type),          intent(in)    :: Time
  type(ice_ocean_boundary_type),  intent(in)    :: Ice_ocean_boundary
  type(ocean_thickness_type),     intent(in)    :: Thickness
  type(ocean_density_type),       intent(in)    :: Dens
  type(ocean_external_mode_type), intent(inout) :: Ext_mode
  type(ocean_prog_tracer_type),   intent(inout) :: T_prog(:)
  type(ocean_velocity_type),      intent(inout) :: Velocity
  real, dimension(isd:,jsd:),     intent(inout) :: pme
  real, dimension(isd:,jsd:),     intent(inout) :: melt
  real, dimension(isd:,jsd:),     intent(inout) :: river
  real, dimension(isd:,jsd:),     intent(inout) :: runoff
  real, dimension(isd:,jsd:),     intent(inout) :: calving
```

```
 ! start of long if-block for use_waterflux true or false.
 if (use_waterflux) then
```

```
&ocean_sbc_nml
      use_waterflux=.true.
      land_model_heat_fluxes=.false.
      temp_restore_tscale=-10.
      salt_restore_tscale=-10.
      max_ice_thickness=8.0                        ! smg 2009/04/13
      avg_sfc_velocity=.true.
      avg_sfc_temp_salt_eta=.true.
      waterflux_tavg=.false. ! false is default setting
      calvingspread=.false.
      runoffspread=.false.
      use_full_patm_for_sea_level=.true.           ! smg 2009/04/09
```

mom5/ocean_core/ocean_sbc.F90

```
! water flux in (kg/m^3)*(m/s) from liquid runoff and calving land ice
do j = jsc_bnd,jec_bnd
   do i = isc_bnd,iec_bnd
      ii = i + i_shift
      jj = j + j_shift
      runoff(ii,jj)  = Ice_ocean_boundary%runoff(i,j)*Grd%tmask(ii,jj,1)
      calving(ii,jj) = Ice_ocean_boundary%calving(i,j)*Grd%tmask(ii,jj,1)
   enddo
enddo
if(use_ideal_runoff) then
    do j=jsc,jec
       do i=isc,iec
          runoff(i,j)  = runoff(i,j)  + ideal_runoff(i,j)
       enddo
    enddo
endif
if(use_ideal_calving) then
    do j=jsc,jec
       do i=isc,iec
          calving(i,j) = calving(i,j) + ideal_calving(i,j)
       enddo
    enddo
endif
if(runoffspread)  call spread_river_horz(runoff)
if(calvingspread) call spread_river_horz(calving)

! river=runoff+calving is the water mass flux
! entering ocean, other than through pme.
! calving is immediately melted, so it is appropriate
! to add it to runoff for purposes of getting a mass
! flux of liquid, from land, into the ocean.
if(waterflux_tavg) then
    do j=jsc,jec
       do i=isc,iec
          river(i,j)       = 0.5*river_taum1(i,j)
          river_taum1(i,j) = runoff(i,j) + calving(i,j)
          river(i,j)       = river(i,j) + 0.5*river_taum1(i,j)
       enddo
    enddo
else
    do j=jsc,jec
       do i=isc,iec
          river(i,j) = runoff(i,j) + calving(i,j)
       enddo
    enddo
endif
```

mom5/ocean_core/ocean_sbc.F90

```
&ocean_sbc_nml
      use_waterflux=.true.
      land_model_heat_fluxes=.false.
      temp_restore_tscale=-10.
```

```
! this be liquid or solid water.
if(land_model_heat_fluxes) then
```

```
else

    ! for cases where the land model does not carry heat flux associated with
    ! the liquid runoff and solid calving ice. We assign a temperature to the
    ! liquid and solid runoff.
    ! Set temperature for liquid runoff water to ocean surface value, but no
    ! less than runoff_temp_min. For other tracers, by default keep them
    ! set to their initial concentration.
    ! For calving temperature, set this equal to the SST.
    do j=jsc,jec
       do i=isc,iec
          T_prog(index_temp)%trunoff(i,j)  = &
             max(runoff_temp_min,T_prog(index_temp)%field(i,j,1,tau))
          T_prog(index_temp)%tcalving(i,j) = &
             T_prog(index_temp)%field(i,j,1,tau)
          T_prog(index_temp)%runoff_tracer_flux(i,j) = &
             Grd%tmask(i,j,1)*T_prog(index_temp)%trunoff(i,j)*runoff(i,j)
          T_prog(index_temp)%calving_tracer_flux(i,j)= &
             Grd%tmask(i,j,1)*T_prog(index_temp)%tcalving(i,j)*calving(i,j)
       enddo
    enddo

endif
```

mom5/ocean_core/ocean_sbc.F90

```
&ocean_sbc_nml
      use_waterflux=.true.
      land_model_heat_fluxes=.false.
      temp_restore_tscale=-10.
```

```
! this be liquid or solid water.
if(land_model_heat_fluxes) then
```

```
else

    ! for cases where the land model does not carry heat flux associated with
    ! the liquid runoff and solid calving ice. We assign a temperature to the
    ! liquid and solid runoff.
    ! Set temperature for liquid runoff water to ocean surface value, but no
    ! less than runoff_temp_min. For other tracers, by default keep them
    ! set to their initial concentration.
    ! For calving temperature, set this equal to the SST.
    do j=jsc,jec
       do i=isc,iec
          T_prog(index_temp)%trunoff(i,j)  = &
             max(runoff_temp_min,T_prog(index_temp)%field(i,j,1,tau))
          T_prog(index_temp)%tcalving(i,j) = &
             T_prog(index_temp)%field(i,j,1,tau)
          T_prog(index_temp)%runoff_tracer_flux(i,j) = &
             Grd%tmask(i,j,1)*T_prog(index_temp)%trunoff(i,j)*runoff(i,j)
          T_prog(index_temp)%calving_tracer_flux(i,j)= &
             Grd%tmask(i,j,1)*T_prog(index_temp)%tcalving(i,j)*calving(i,j)
       enddo
    enddo

endif
```

```
do n=1,num_prog_tracers
   if (T_prog(n)%name == 'temp') index_temp = n
   if (T_prog(n)%name == 'salt') index_salt = n
enddo
```

```
! compute temperature and salinity of "river" according to
! temperature and salinity of calving and runoff.
! allow for possibility of runoff and/or calving to be negative,
! which may exist for certain land models that "suck" water from
! the oceans to address limitations in their formulation.
! We do not include such negative points when computing triver.
! Also compute the salinity flux associated with liquid and solid runoff.
do j=jsc,jec
   do i=isc,iec
      tmp_runoff = max(0.0,runoff(i,j))
      tmp_calving= max(0.0,calving(i,j))
      T_prog(index_temp)%triver(i,j) = Grd%tmask(i,j,1)    &
           *(tmp_runoff *T_prog(index_temp)%trunoff(i,j)   &
            +tmp_calving*T_prog(index_temp)%tcalving(i,j)) &
           /(epsln + tmp_runoff + tmp_calving)

      T_prog(index_salt)%triver(i,j) = Grd%tmask(i,j,1)    &
           *(tmp_runoff *T_prog(index_salt)%trunoff(i,j)   &
            +tmp_calving*T_prog(index_salt)%tcalving(i,j)) &
           /(epsln + tmp_runoff + tmp_calving)
   enddo
enddo
```

mom5/ocean_core/ocean_model.F90 – ocean_model_init

```
T_prog => ocean_prog_tracer_init(Grid, Thickness, Ocean_options, Domain, Time, Time_steps, &
                                 num_prog_tracers, vert_coordinate_type, have_obc,         &
                                 cmip_units, use_blobs, debug=debug)
write(*,*) 'Moon: after initializing prognostic tracers in ocean_model_init'
```

```
! fill the field table entries for the prognostic tracers
n = 0
do while (fm_loop_over_list('/ocean_mod/prog_tracers', name, typ, ind))  !{

   if (typ .ne. 'list') then  !{

       call mpp_error(FATAL, trim(error_header) // ' ' // trim(name) // ' is not a list')

   else  !}{

       n = n + 1  ! increment the array index

       if (n .ne. ind) then  !{
           write (stdoutunit,*) trim(warn_header), ' Tracer index, ', ind,   &
                ' does not match array index, ', n, ' for ', trim(name)
       endif  !}

       ! save the name
       T_prog(n)%name = name

       if (.not. fm_change_list('/ocean_mod/prog_tracers/' // trim(name))) then  !{
           call mpp_error(FATAL, trim(error_header) // ' Problem changing to ' // trim(name))
       endif  !}

       caller_str = 'ocean_tracer_mod(ocean_prog_tracer_init)'

       ! save the units
       T_prog(n)%units = fm_util_get_string('units', caller = caller_str, scalar = .true.)

       ! save the type
       T_prog(n)%type = fm_util_get_string('type', caller = caller_str, scalar = .true.)

       ! save the longname
       T_prog(n)%longname = fm_util_get_string('longname', caller = caller_str, scalar = .true.)

       ! save the conversion
       T_prog(n)%conversion = fm_util_get_real('conversion', caller = caller_str, scalar = .true.)

       ! save the offset
       T_prog(n)%offset = fm_util_get_real('offset', caller = caller_str, scalar = .true.)

       ! get the min and max of the tracer
       T_prog(n)%min_tracer = fm_util_get_real('min_tracer', caller = caller_str, scalar = .true.)
       T_prog(n)%max_tracer = fm_util_get_real('max_tracer', caller = caller_str, scalar = .true.)

       ! get the min and max of the range for analysis
       T_prog(n)%min_range = fm_util_get_real('min_range', caller = caller_str, scalar = .true.)
       T_prog(n)%max_range = fm_util_get_real('max_range', caller = caller_str, scalar = .true.)

       ! get the flux unit
       T_prog(n)%flux_units = fm_util_get_string('flux_units', caller = caller_str, scalar = .true.)

       ! get the min and max of the flux range for analysis
       T_prog(n)%min_flux_range = fm_util_get_real('min_flux_range', caller = caller_str, scalar = .true.)
       T_prog(n)%max_flux_range = fm_util_get_real('max_flux_range', caller = caller_str, scalar = .true.)

       ! save the restart file
       T_prog(n)%restart_file = fm_util_get_string('restart_file', caller = caller_str, scalar = .true.)
```

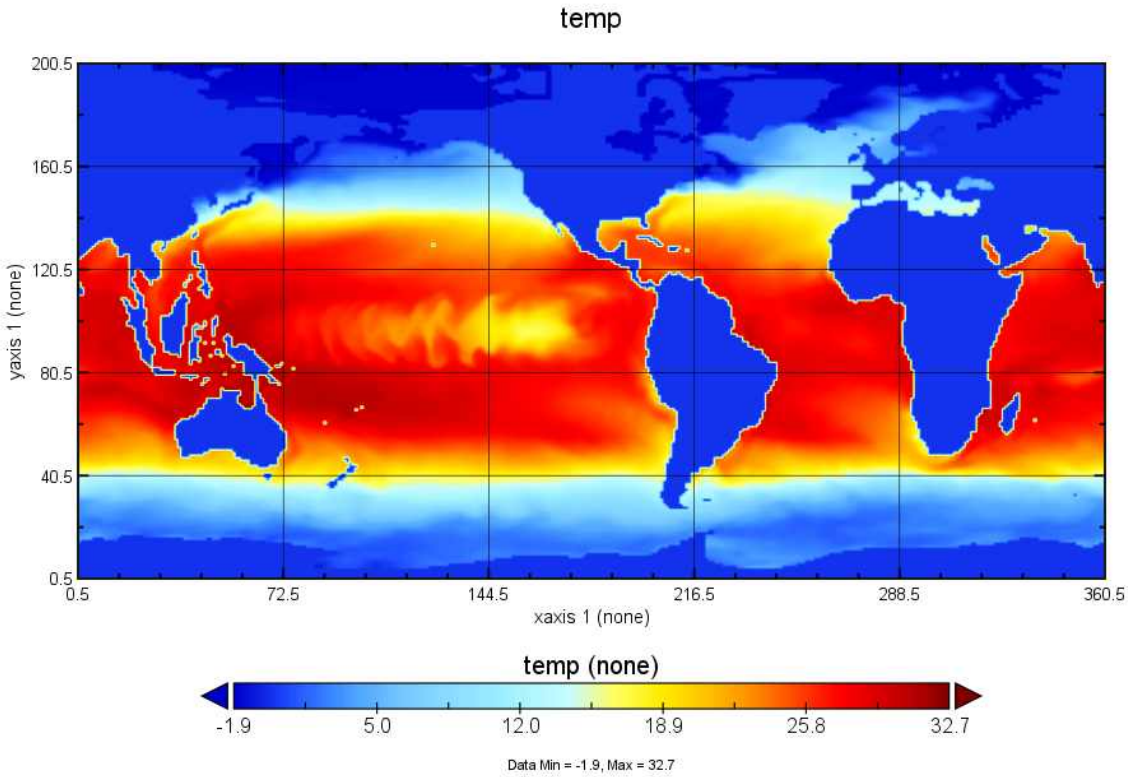
field_table

```
###  MDPPM for T,S
"prog_tracers","ocean_mod","temp"
horizontal-advection-scheme = mdppm
vertical-advection-scheme = mdppm
restart_file  = ocean_temp_salt.res.nc
ppm_hlimiter=2
ppm_vlimiter=2
/
"prog_tracers","ocean_mod","salt"
horizontal-advection-scheme = mdppm
vertical-advection-scheme = mdppm
restart_file  = ocean_temp_salt.res.nc
ppm_hlimiter=2
ppm_vlimiter=2
/
```

sdout

```
Moon: name =
/ocean_mod/prog_tracers/temp

Moon: path =
/ocean_mod/prog_tracers/

Moon: base = temp
Moon: name =
/ocean_mod/prog_tracers/salt

Moon: path =
/ocean_mod/prog_tracers/
```

0.000000000000000E+000	0.000000000000000E+000	0.000000000000000E+000
0.000000000000000E+000	0.000000000000000E+000	2.85471261226112
2.19911459165442	2.16884517103913	2.43385385090545
2.79193617138694	3.09232823439386	3.37071954284793
3.64197325288574	3.92786972407092	4.21962504434882
4.50535513056411	4.82194201456334	5.12504454604085
5.55351831439523	6.15572228185554	6.87410549328929
7.55539767768729	8.06335516428351	8.45618355197401
8.81860920658984	9.25467516298960	9.70822568000174
10.2499830403709	10.8423164590618	11.5326756232167
12.2915596338073	13.1105391351492	13.9501070670030
14.7837174356663	15.6016039700022	16.4487551376949
17.2702031484982	18.1987437912512	19.1305003462149
20.1867495337862	21.2142059012643	22.3040050776155
23.1732767365629	23.9645962177738	24.6054861910725
25.3288847137375	26.0414193671386	26.7746521496148
27.1322048504650	27.2440096287061	27.2204158706979
27.1624686876466	26.9843449838115	26.7351258355138
26.5168005970708	26.3361530196877	26.1450915941478
25.9653720910091	25.7826927674832	25.5970227179828
25.4561533422212	25.3609709095897	25.3479002084348
25.3843302283378	25.4850300251944	25.7192441970630
25.9228960980400	26.1276155477210	26.3466991071137
26.5635221537306	26.7973621180587	26.9982804173920
27.1153987730512	27.1940105018469	27.2252022217320
27.2755481207182	27.3367268475101	27.3713891891445
27.3100924440174	27.0969768280546	26.6153879336615
25.9118583094689	25.4338692280821	25.3262592732462
25.2926111051189	25.1927889449260	25.1130382773378
25.2299559138350	25.8920997284774	26.3541529418242
26.4403406009900	26.2626836948727	26.1040612144184
26.1529818186842	26.3672562871602	26.4854812036659
26.5096330222714	26.5079050505920	26.5274456616852
26.5769888750726	26.4597354946691	25.8147945974319
25.1405853541021	24.7395017373996	24.2417303604299
24.2104828275052	24.0029962824540	24.0680381648334
24.6624969738484	25.6404800172557	26.3865453422639
26.7161756135383	26.9612909316780	27.4489761060881
28.0019126048599	28.1746113483908	28.2730627827845
28.3207977207095	28.1186611650483	27.7957655702083
27.4935368655231	0.000000000000000E+000	0.000000000000000E+000
0.000000000000000E+000	0.000000000000000E+000	0.000000000000000E+000



./ocean_shared/generic_tracers/generic_tracer_utils.F90

```
subroutine g_tracer_flux_init(g_tracer)
  type(g_tracer_type), pointer :: g_tracer


  !=====================================================================
  !Get coupler flux indices
  !=====================================================================
  !
  !For each kind of flux allocate the appropriate arrays only if that flux
  !indicated to exist for the tracer.

  if(g_tracer%flux_gas) then
     g_tracer%flux_gas_ind  = aof_set_coupler_flux(g_tracer%flux_gas_name,                &
          flux_type         = g_tracer%flux_gas_type,                                     &
          implementation    = 'ocmip2',                                                   &
          mol_wt            = g_tracer%flux_gas_molwt,                                    &
          param             = g_tracer%flux_gas_param,                                    &
          ice_restart_file  = g_tracer%ice_restart_file,                                  &
          ocean_restart_file= g_tracer%flux_gas_restart_file                              &
          )
  endif

  if(g_tracer%flux_runoff) then
     g_tracer%flux_runoff_ind  = aof_set_coupler_flux(g_tracer%flux_runoff_name,          &
          flux_type            = 'land_sea_runoff',                                      &
          implementation       = 'river',                                                &
          param                = g_tracer%flux_param,                                    &
          ice_restart_file     = g_tracer%ice_restart_file                               &
          )
  endif
```

./shared/coupler/atmos_ocean_fluxes.F90

```
subroutine atmos_ocean_fluxes_calc(gas_fields_atm, gas_fields_ice,      &
     gas_fluxes, seawater)  !{

!
```

```
elseif (gas_fluxes%bc(n)%flux_type .eq. 'land_sea_runoff') then  !}{

  if (gas_fluxes%bc(n)%param(1) .le. 0.0) then
    write (error_string, '(1pe10.3)') gas_fluxes%bc(n)%param(1)
    call mpp_error(FATAL, ' Bad parameter (' // trim(error_string) //       &
         ') for land_sea_runoff for ' // trim(gas_fluxes%bc(n)%name))
  endif

  length = size(gas_fluxes%bc(n)%field(1)%values(:))

  if (gas_fluxes%bc(n)%implementation .eq. 'river') then  !{

    do i = 1, length  !{
      if (seawater(i) == 1) then  !{
        gas_fluxes%bc(n)%field(ind_flux)%values(i) =        &
             gas_fields_atm%bc(n)%field(ind_deposition)%values(i) / gas_fluxes%bc(n)%param(1)
      else  !}{
        gas_fluxes%bc(n)%field(ind_flux)%values(i) = 0.0
      endif  !}
```

./coupler/flux_exchange.F90

```
subroutine sfc_boundary_layer ( dt, Time, Atm, Land, Ice, Land_Ice_Atmos_Boundary )

  real                    intent(in)    :: dt
```

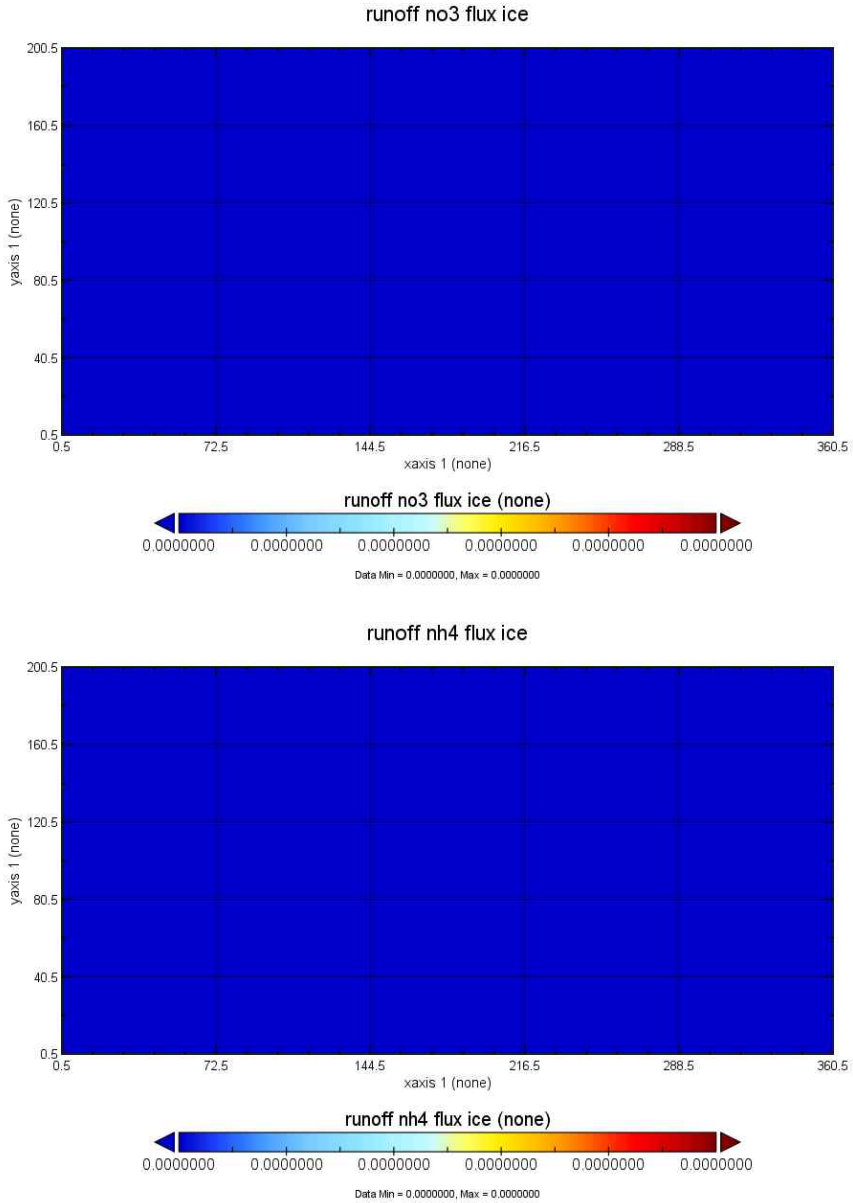
```
! Combine explicit ocean flux and implicit land flux of extra flux fields.

  ! Calculate ocean explicit flux here

  call atmos_ocean_fluxes_calc(ex_gas_fields_atm, ex_gas_fields_ice, ex_gas_fluxes, ex_seawater)

  ! The following statement is a concise version of what's following and worth
  ! looking into in the future.
  ! ex_flux_tr(:,itracer) = ex_gas_fluxes%bc(itracer_ocn)%field(ind_flux)%values(:)
  ! where(ex_seawater.gt.0) ex_flux_tr(:,itracer) = F_ocn
  print*, 'synim aaa'
  do n = 1, ex_gas_fluxes%num_bcs  !{
    if (ex_gas_fluxes%bc(n)%atm_tr_index .gt. 0) then  !{
      m = tr_table_map(ex_gas_fluxes%bc(n)%atm_tr_index)%exch
      do i = 1, size(ex_seawater(:))  !{
         if (ex_land(i)) cycle  ! over land, don't do anything
         ! on ocean or ice cells, flux is explicit therefore we zero derivatives.
         ex_dfdtr_atm(i,m)  = 0.0
         ex_dfdtr_surf(i,m) = 0.0
         if (ex_seawater(i)>0) then
            ! jgj: convert to kg co2/m2/sec for atm
            ex_flux_tr(i,m)    = ex_gas_fluxes%bc(n)%field(ind_flux)%values(i) * ex_gas_fluxes%bc(n)%mol_wt * 1.0e-03
         else
            ex_flux_tr(i,m) = 0.0 ! pure ice exchange cell
        endif  !}
      enddo  !} i
    endif  !}
    print*, n, ex_gas_fluxes%bc(n)%mol_wt
  enddo  !} n

  print*, size(ex_flux_tr,1), size(ex_flux_tr,2)
```

src/coupler/coupler_main.F90

```
if ( Ice%pe ) then
  call mpp_set_current_pelist(Ice%pelist)
  allocate(Ice_bc_restart(Ice%ocean_fluxes%num_bcs))
  allocate(ice_bc_restart_file(Ice%ocean_fluxes%num_bcs))
  do n = 1, Ice%ocean_fluxes%num_bcs  !{
    if(Ice%ocean_fluxes%bc(n)%num_fields .LE. 0) cycle
    filename = trim(Ice%ocean_fluxes%bc(n)%ice_restart_file)
    do l = 1, num_ice_bc_restart
       if(trim(filename) == ice_bc_restart_file(l)) exit
    end do
    if(l>num_ice_bc_restart) then
       num_ice_bc_restart = num_ice_bc_restart + 1
       ice_bc_restart_file(l) = trim(filename)
    end if
    filename = 'INPUT/'//trim(filename)
    other_fields_exist = .false.
    do m = 1, Ice%ocean_fluxes%bc(n)%num_fields  !{
      fieldname = trim(Ice%ocean_fluxes%bc(n)%field(m)%name)
      id_restart = register_restart_field(Ice_bc_restart(l), ice_bc_restart_file(l), &
                   fieldname, Ice%ocean_fluxes%bc(n)%field(m)%values, Ice%domain    )
      if (field_exist(filename, fieldname, Ice%domain) ) then
        other_fields_exist = .true.
        write (outunit,*) trim(note_header), ' Reading restart info for ',         &
             trim(fieldname), ' from ',  trim(filename)
        print*, 'synim 111', filename
        print*, 'synim 222', fieldname
        call read_data(filename, fieldname, Ice%ocean_fluxes%bc(n)%field(m)%values, Ice%domain)
        print*, 'synim 333', n, m
        print*, Ice%ocean_fluxes%bc(n)%field(m)%values
      elseif (other_fields_exist) then
        call mpp_error(FATAL, trim(error_header) // ' Couldn''t find field ' //     &
             trim(fieldname) // ' in file ' //trim(filename))
      endif
    enddo  !} m
  enddo  !} n
```

src/coupler/coupler_main.F90

filename		INPUT/ice_topaz.res.nc
field name	n = 1, m=1	o2_flux_flux_ice
	n = 1, m=2	o2_flux_deltap_ice
	n = 1, m=3	o2_flux_kw_ice
	n = 2, m=1	**runoff_no3_flux_ice**
	n = 3, m=1	wet_dep_no3_flux_ice
	n = 4, m=1	dry_dep_no3_flux_ice
	n = 5, m=1	**runoff_nh4_flux_ice**
	n = 6, m=1	wet_dep_nh4_flux_ice
	n = 7, m=1	dry_dep_nh4_flux_ice
	n = 8, m=1	**runoff_lith_flux_ice**
	n = 9, m=1	wet_dep_lith_flux_ice
	n = 10, m=1	dry_dep_lith_flux_ice
	n = 11, m=1	**runoff_ldon_flux_ice**
	n = 12, m=1	**runoff_fed_flux_ice**
	n = 13, m=1	wet_dep_fed_flux_ice
	n = 14, m=1	dry_dep_fed_flux_ice
	n = 15, m=1	co2_flux_flux_ice
	n = 15, m=2	co2_flux_deltap_ice
	n = 15, m=3	co2_flux_kw_ice
	n = 16, m=1	**runoff_dic_flux_ice**
	n = 17, m=1	**runoff_alk_flux_ice**



src/coupler/coupler_main.F90