For my previous PR on MOM6 code update, the RT results changed which all of us felt unusual, so I did a lot of debug work and with help of GFDL, we finally found out that there is indeed a bug in the code, Bob forgot to add "halo" in one of the equator of state code. You may see line 411

https://github.com/jiandewang/MOM6/blob/3850530366707a9fd5676b97fe63d53ff52b2c5b/src/parameterizations/vertical/MOM_full_convection.F90

With this bug fixing code, I re-ran the RT test (totally 30), this is what I got:

(1) 10 cases passed

(2) 18 cases failed in ocean restart file comparison, they all failed in the final comparison of ocean restart files, all other files being compared are identical. This is what shows in one of the log file:

Checking test 030 cpld_fv3_ccpp_mom6_cice_cmeps_6h_debug results ....

Comparing phyf006.tile1.nc ....OK

Comparing phyf006.tile2.nc ....OK

Comparing phyf006.tile3.nc ....OK

Comparing phyf006.tile4.nc ....OK

Comparing phyf006.tile5.nc ....OK

Comparing phyf006.tile6.nc ....OK

Comparing dynf006.tile1.nc ....OK

Comparing dynf006.tile2.nc ....OK

Comparing dynf006.tile3.nc ....OK

Comparing dynf006.tile4.nc ....OK

Comparing dynf006.tile5.nc ....OK

Comparing dynf006.tile6.nc ....OK

Comparing RESTART/coupler.res ....OK

Comparing RESTART/fv_core.res.nc ....OK

Comparing RESTART/fv_core.res.tile1.nc ....OK

Comparing RESTART/fv_core.res.tile2.nc ....OK

Comparing RESTART/fv_core.res.tile3.nc ....OK

Comparing RESTART/fv_core.res.tile4.nc ....OK

Comparing RESTART/fv_core.res.tile5.nc ....OK

Comparing RESTART/fv_core.res.tile6.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile1.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile2.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile3.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile4.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile5.nc ....OK

Comparing RESTART/fv_srf_wnd.res.tile6.nc ....OK

Comparing RESTART/fv_tracer.res.tile1.nc ....OK

Comparing RESTART/fv_tracer.res.tile2.nc ....OK

Comparing RESTART/fv_tracer.res.tile3.nc ....OK

Comparing RESTART/fv_tracer.res.tile4.nc ....OK

Comparing RESTART/fv_tracer.res.tile5.nc ....OK

Comparing RESTART/fv_tracer.res.tile6.nc ....OK

Comparing RESTART/phy_data.tile1.nc ....OK

Comparing RESTART/phy_data.tile2.nc ....OK

Comparing RESTART/phy_data.tile3.nc ....OK

Comparing RESTART/phy_data.tile4.nc ....OK

Comparing RESTART/phy_data.tile5.nc ....OK

Comparing RESTART/phy_data.tile6.nc ....OK

Comparing RESTART/sfc_data.tile1.nc ....OK

Comparing RESTART/sfc_data.tile2.nc ....OK

Comparing RESTART/sfc_data.tile3.nc ....OK

Comparing RESTART/sfc_data.tile4.nc ....OK

Comparing RESTART/sfc_data.tile5.nc ....OK

Comparing RESTART/sfc_data.tile6.nc ....OK

Comparing RESTART/MOM.res.nc ....NOT OK

Comparing RESTART/MOM.res_1.nc ....NOT OK

Comparing RESTART/MOM.res_2.nc ....NOT OK

Comparing RESTART/MOM.res_3.nc ....NOT OK

Comparing RESTART/iced.2016-10-03-21600.nc ....OK

Comparing ufs.s2s.cpl.r.2016-10-03-21600.nc ....OK

Test 030 cpld_fv3_ccpp_mom6_cice_cmeps_6h_debug FAIL

The reason for the different in ocean restart files is that MOM6 saves a hexadecimal string "checksum" value in the restart file header, with this version of code there are some minor changes in restart code, thus the checksum values changed compared with EMC current RT results, but all the variables values are the same. Note checksum values will not be involved in model integration, it is a checker which is being read in the very beginning.

I compared the ocean output with what is in the current RT directory, they are identical. I also compared the values on ocean restart files, and they are identical to RT files. Besides this I also did "ncdump" for all the restart files, and compared with current RT files, the only differences are in header checksum. Here is an example of what I got in one of the "diff" between two dumped text files:

33c33

<   Temp:checksum = "E70BB5979ABE97A1" ;

>   Temp:checksum = "FFFFFFFF9ABE97A1" ;

37c37

<   Salt:checksum = "E4E47ACE6D13A418" ;

>   Salt:checksum = "   6D13A418" ;

41c41

<   h:checksum = "411813FC4AF2D5CF" ;

>   h:checksum = "   4AF2D5CF" ;

45c45

<   u:checksum = "D1030A5996ABE2A4" ;

>   u:checksum = "FFFFFFFF96ABE2A4" ;

You may see all the "diff" results on HERA at
/scratch2/NCEPDEV/climate/Jiande.Wang/zzz-Bob-fix/NC-diff/Diff-txt

Note what I did is "ncdump", not "ncdump -c", in other words I compared the whole body of the restart files (~34 million lines for one restart file), not the header only.

As I mentioned in the beginning, there are 10 cases that passed. The reason they got passed is because in the settings they do not compare ocean restart files. Otherwise they would have been marked as fail.

07/01 Update on checksum issue:

I was able to locate the exact commit from GFDL MOM6 commit history which caused the changing of checksum values. It is “22215cb0b7781, Apr 14, Internal field index rotation” by Marshall Ward, and this got confirmed by GFDL side. So the checksum issue can be cleared.

The following explains what is checksum (By Marshall Ward)

I believe that FMS takes the number (in floating point format) and recasts it into an integer, and then just sums all of these integers together. The result is a large arbitrary hex number which gets printed as a string.

It just does something like this for some field f:

```
f_int = transfer(f, int)   ! Cast float to int

chk = sum(f_int, mask=land_mask)   ! Sum ints on PE

call mpp_sum(chk)  ! Sum across PEs

print('Z16', chk)  ! Report as a hexadecimal code
```

It's not a perfect checksum, but it is a decent way to confirm that what you are using has not changed (e.g. bit corruption). Mostly I think it's just been around so long that people have gotten used to it.

The number is not supposed to change if you change PE layouts, but it can change on different platforms. But if you're going from, say, Intel to Intel and using the same compilers, then it should not change.

In the case of the index rotation code, it is possible that a -0 was changed into a +0 (or vice versa) which would change the bit representation of certain numbers. This could modify the checksums, for example, but not the actual values.

Now the puzzle comes:

The remaining two cases failed for real. They are cmep benchmark cold runs:

(1) cpld_fv3_ccpp_384_mom6_cice_cmeps_cold_bmark_rt

(2) cpld_fv3_ccpp_384_mom6_cice_cmeps_ww3_cold_bmark_rt

The run directory and log files can be found on HERA at

/scratch1/NCEPDEV/stmp2/Jiande.Wang/MOM6-update-RT-run/RT-run-rt_28588

As I know little on cmep, here I need Denise's involvement to help me figure out the root cause. What puzzled me are:

(1) the corresponding cmep benchmark warm runs (cpld_fv3_ccpp_384_mom6_cice_cmeps_1d_bmark_rt) are fine (here fine I mean all results are identical except ocean restart files diffs in checksum), but cold start runs failed

(2) the corresponding nems benchmark cold runs (cpld_fv3_ccpp_384_mom6_cice_cold_bmark_rt) are fine.

MOM6 code is at https://github.com/jiandewang/MOM6/tree/feature/update-to-GFDL-20200515

ufs-s2s-model branch is at https://github.com/jiandewang/ufs-s2s-model/tree/feature/MOM6-20200515-bug-fixing

I have a working copy on HERA at

/scratch2/NCEPDEV/climate/Jiande.Wang/zzz-Bob-fix/s2s-MOM6-update-JW

All RT runs results are on HERA at

/scratch1/NCEPDEV/stmp2/Jiande.Wang/MOM6-update-RT-run/RT-run-rt_28588

Note I added “RESTART_CHECKSUMS_REQUIRED= F” in MOM_input, otherwise those runs which use CPC ocean DA as IC will not start. This parameter is not used in model integration thus has no impact on final results.

All current ufs-s2s-model develop RT runs are on HERA at:

/scratch1/NCEPDEV/stmp2/Jiande.Wang/s2s-EMC_HEAD-RT-run/rt_234160

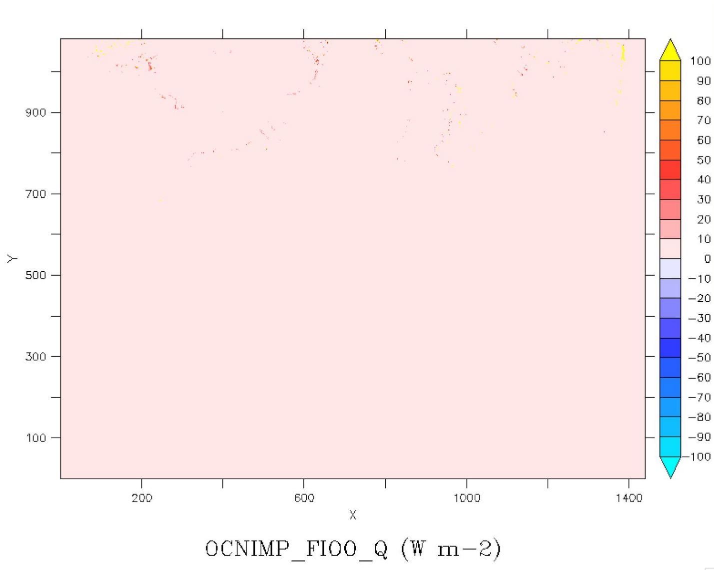
Denise's notes:

1) With the CMEPS mediator, it is possible to insert the writing of the mediator coupler files at different points in the run sequence. So the first thing was to insert a history write into the run sequence here:

```
# CMEPS cold run sequence

runSeq::
  @1800
    @450
      MED med_phases_prep_atm
      MED -> ATM :remapMethod=redist
      ATM
      ATM -> MED :remapMethod=redist
      MED med_phases_prep_ice
      MED -> ICE :remapMethod=redist
      MED med_phases_history_write
      ICE
      ICE -> MED :remapMethod=redist
      MED med_fraction_set
      MED med_phases_prep_ocn_map
      MED med_phases_prep_ocn_merge
      MED med_phases_prep_ocn_accum_fast
    @
    MED med_phases_prep_ocn_accum_avg
    MED -> OCN :remapMethod=redist
    OCN
    OCN -> MED :remapMethod=redist
    MED med_phases_restart_write
@
::
```

Mediator history writes here will be after the ocn has been initialized and right after those values have been sent to ice for the first time (but before ice actually runs). Comparing the coupler history at the first time step (450s) between the new MOM6 vs existing MOM6 showed one field was different between the two MOM versions: the freeze-melt potential (difference shown below). This creates different ice fields returning from the ice at the first timestep which then propagates. This explains why the updated MOM6 cold start with CMEPS does not reproduce.



2) Using NEMS versions of the code (with updated MOM6, existing MOM6), I then investigated why the NEMS version apparently reproduces existing baseline. I turned on field-dumping for the NEMS versions and created a similar difference between the runs using the field_init_ocn_export_2012-01-01T00:00:00.nc field dumps. In fact, the exact same difference shown above appears in the NEMS cases. So why does the NEMS version reproduce?

In the NEMS mediator version, the models are not initialized correctly. The ice model does not obtain non-zero values from the ocean until the ocean takes it's first timestep. This means the ice model takes it's first 4 timesteps with "0.0" for all the fields being sent by the ocean (even though ocean has actual values).

In the CMEPS versions, datainitialize occurs correctly and the ice model has valid values for those first four timesteps:

Values *imported* by ice at one location which has changed values for freeze-melt over the 8 timesteps of the cold start:

NEMS: Column 1 (current MOM6), Column 2 (updated MOM6)

```
 Column  1: FREEZING_MELTING_POTENTIAL[D=nems.frzmlt] is FREEZING_MELTING_POTENTIAL
 Column  2: FREEZING_MELTING_POTENTIAL[D=nems_jw.frzmlt] is FREEZING_MELTING_POTENTIAL
          FREEZING  FREEZING_MELTING_POTENTIAL
1   / 1:      0.      0.
2   / 2:      0.      0.
3   / 3:      0.      0.
4   / 4:      0.      0.
5   / 5: -24106. -24106.
6   / 6: -24106. -24106.
7   / 7: -24106. -24106.
8   / 8: -24106. -24106.
```

CMEPS: Column 1 (current MOM6), Column 2 (updated MOM6)

```
 Column  1: FREEZING_MELTING_POTENTIAL[D=cmeps.frzmlt] is FREEZING_MELTING_POTENTIAL
 Column  2: FREEZING_MELTING_POTENTIAL[D=cmeps_jw.frzmlt] is FREEZING_MELTING_POTENTIAL
       FREEZING  FREEZING_MELTING_POTENTIAL
1   / 1:     13. -27934.
2   / 2:     13. -27934.
3   / 3:     13. -27934.
4   / 4:     13. -27934.
5   / 5: -24427. -22953.
6   / 6: -24427. -22953.
7   / 7: -24427. -22953.
8   / 8: -24427. -22953.
```

In the NEMS case, the ice_fraction exported by the ice is the same for either MOM version:

```
 Column  1: ICE_FRACTION[D=nems.ice] is ICE_FRACTION
 Column  2: ICE_FRACTION[D=nems_jw.ice] is ICE_FRACTION
       ICE_FRAC  ICE_FRACTION
1   / 1:  0.9404  0.9404
2   / 2:  0.9388  0.9388
3   / 3:  0.9372  0.9372
4   / 4:  0.9355  0.9355
5   / 5:  0.9323  0.9323
6   / 6:  0.9288  0.9288
7   / 7:  0.9252  0.9252
8   / 8:  0.9216  0.9216
```

In CMEPS, because the freeze-melt is both non-zero (and different), the ice fraction changes between the two MOM6 versions:

```
 Column  1: ICE_FRACTION[D=cmeps.ice] is ICE_FRACTION
 Column  2: ICE_FRACTION[D=cmeps_jw.ice] is ICE_FRACTION
       ICE_FRAC  ICE_FRACTION
1   / 1:  0.9390  0.9387
2   / 2:  0.9357  0.9350
3   / 3:  0.9327  0.9316
4   / 4:  0.9297  0.9282
5   / 5:  0.9263  0.9247
6   / 6:  0.9227  0.9212
7   / 7:  0.9192  0.9177
8   / 8:  0.9157  0.9141
```