NATIONAL WEATHER SERVICE

Asynchronous IO component in UFS weather model

Jun Wang, Dusan Jovic, Denise Worthen, Raffaele Montuoro, Bin Liu, Jiande Wang
Ann Tsay, Dan Rosen, Bill Sacks, Gerhard Theurich, Jim Edwards

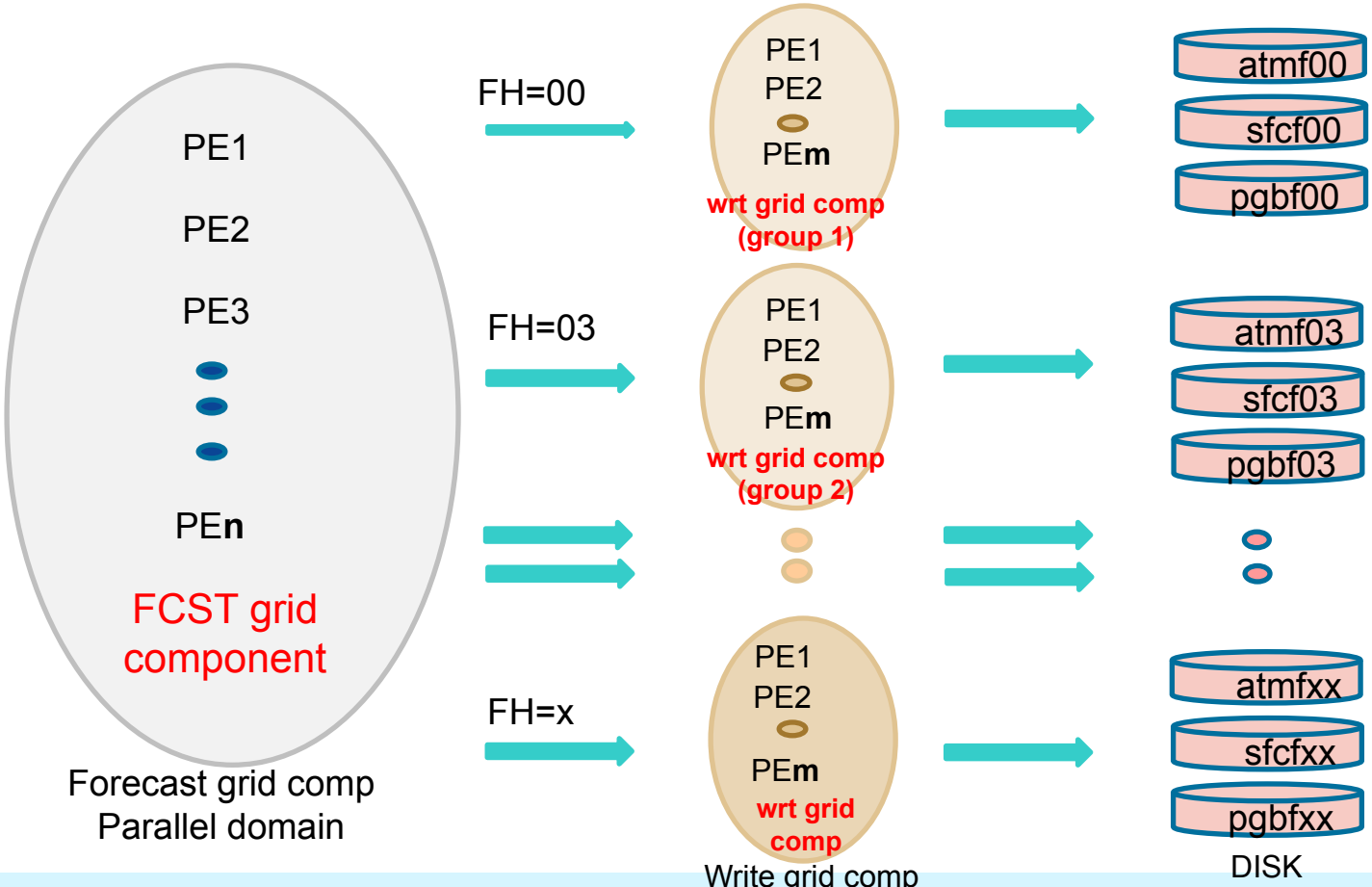
Outline

- Current capability of asynchronous write grid component in UFS
- Initial effort on developing generic write grid component
- Future plans

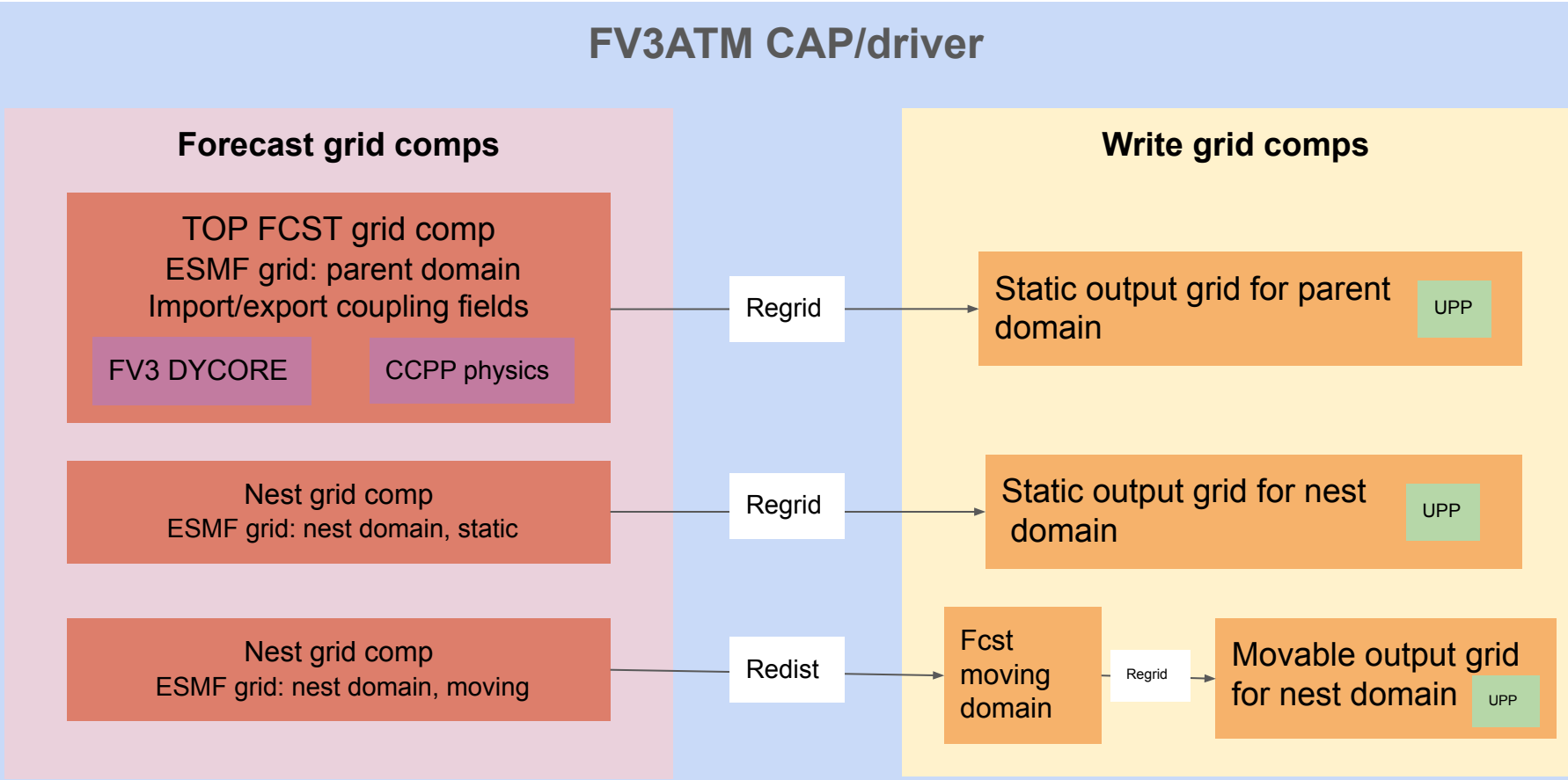
Develop asynchronous IO component in UFS weather model

- **IO performance is critical to operational models.**
  - There is a restriction for the model to complete within an operational window.
  - Outputting model history files takes time, and these files need to show up at specific time.
  - Model resolutions have been increased and the amount of output data has been increased significantly.
- **Asynchronous write grid component**
  - Separate forecast integration from writing output files in the atmosphere component FV3TAM
  - First implemented with inline post in UFS weather model in 2016 for GFSv15
  - Has been used in operational models including GEFSv12 and GFSv16
  - The capability has been extended for moving nests for HAFSv1

Parallelization of FV3ATM write grid component



Extending write grid component capability for HAFSv1



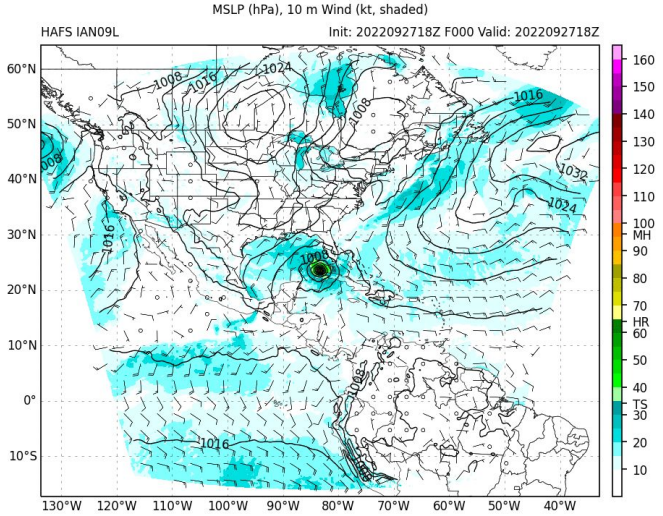
IO improvement in GFS implementations

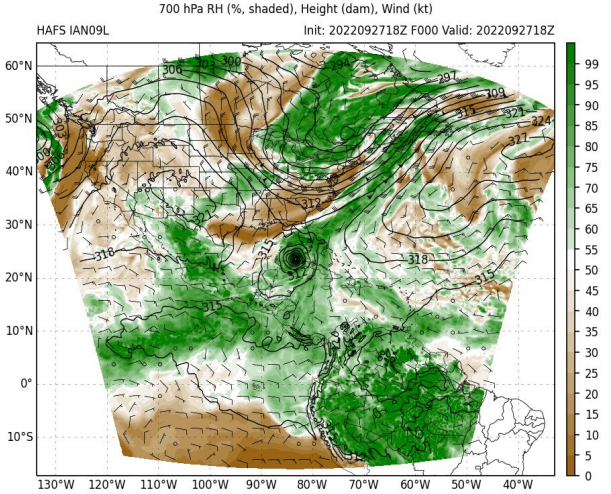
C768L127f cst output	Nemsio No compression	Netcdf No compression	Netcdf Lossless (deflate=1,nbit=0)	Netcdf Lossy (deflate =1, nbit=20)	Netcdf Lossy(deflate=1,nbit=14)	Netcdf Lossy (deflate=1, nbits=14),parallel writing, default decomposition chunksize	Netcdf Lossy (deflate=1, nbits=14),parallel writing Layer chunkcsize
A 3D file size (total fcst)	33.6GB (7TB)	33.6GB (7TB)	23.6GB (5TB)	13.5GB (2.8TB)	6.3GB (1.3TB)	6.3GB (1.3TB)	6.3GB (1.3TB)
Write Time	79s	300s	960s	680s	400s	43s	34s

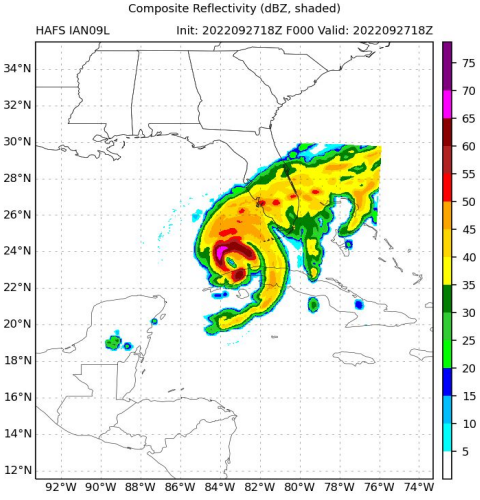
experiments	C96L64 (6 tasks)	C192L64 (12 tasks)	C768L127 (84 tasks)
Single master file size	51MB	180MB	2.5GB
Inline post time	4s	7s	39s
Offline post time	12s	17s	211s

Asynchronized IO component in HAFSv1

- Asynchronous write grid components are used in HAFS v1 with moving nests
- Inline post capability is extended to HAFS multiple grid moving nest applications.
- History files and post processing products are output independently on each grid

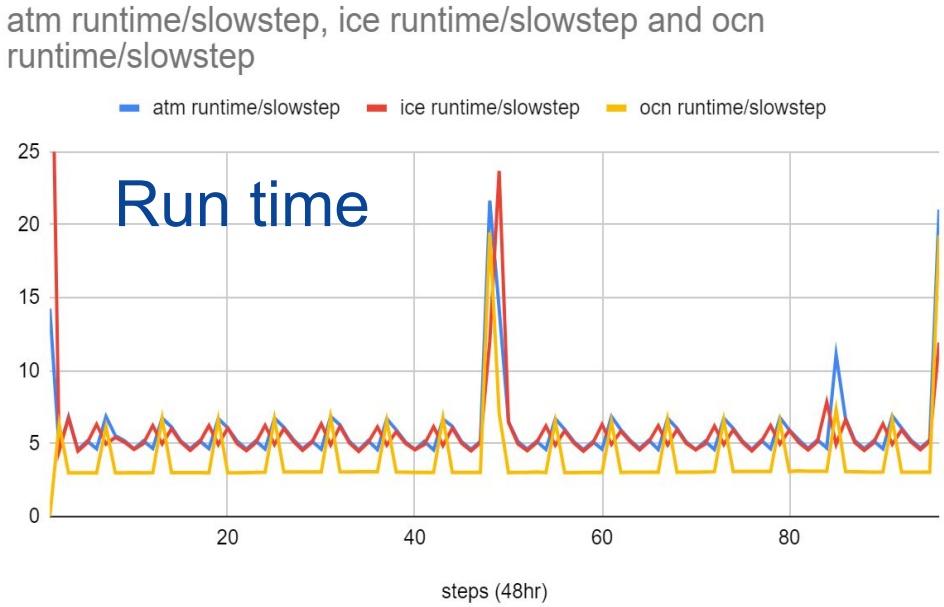


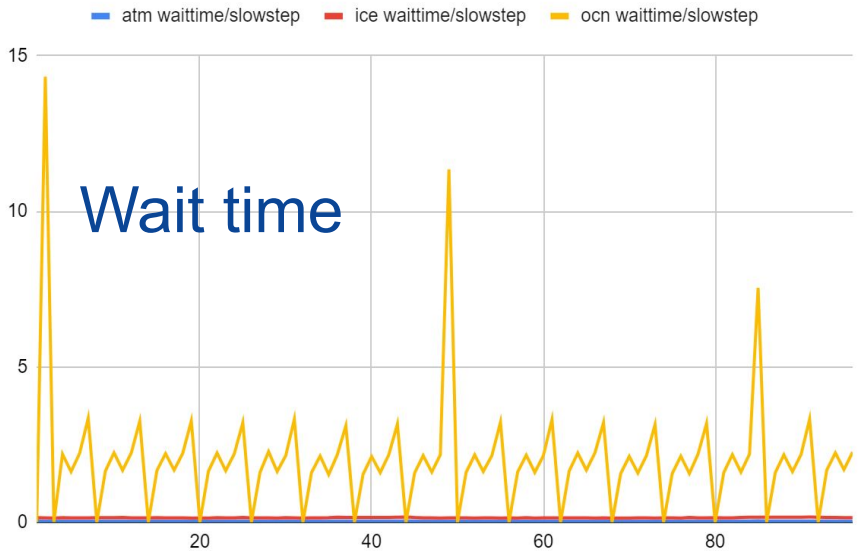




Ian forecast images: Lin Zhu

Challenges: IO impact in UFS coupled configurations:





GFSv17 HR2 S2S C768mx25: atm: 32x32,8x64,ocn/ice240

To develop generic asynchronous IO for earth modeling components

- Communication with the modeling community to collect requirements on asynchronous output
- The discussion is focusing on developing a asynchronized generic write grid component (O, not I)
  - The write component can be used for multiple model components (generic)
  - A model component can communicate with one or more write components
  - The write component receives self describing output fields from a model component and write out these fields into files.
  - Output grids can be defined on the write component
  - The data can be redistributed or remapped from a model component to the write component
  - The write component can execute other data process code, such as inline post.
  - The write grid component can output files at certain forecast time or in a specific frequency

Design discussion

- **Three approaches were discussed:**
  - A NUOPC component
    - Write grid component is implemented as a NUOPC component
  - An ESMF component
    - Write grid component is implemented as an ESMF component
  - PIO
    - Asynchronized Writing is implemented through PIO

Initial testing with generic IO component

Initial testing has been conducted to set up the NUOPC write component with CICE6:

- UFSIO based on the SWIO and COMIO developed by Raffaele Montuoro were adopted
- CICE6 output fields are specified in the IO configuration file to avoid issues with component handshaking
- ESMF fields are built on the CICE6 native grid for the variables in the restart files and added to CICE6 export state
- Temporary workaround is made to pass some attributes required in the restart files (e.g. coordinate dimension names etc)
- Run sequence is updated.
- Restart files are written out from the UFSIO component with data values identical to those in the current restart files.

Run Sequence:

```
runSeq::
@43200    # 12h
  @@[coupling_interval_slow_sec]
    MED med_phases_prep_ocn_avg
    MED -> OCN :remapMethod=redist
    OCN
    @@[coupling_interval_fast_sec]
      MED med_phases_prep_atm
      MED med_phases_prep_ice
      MED med_phases_prep_wav_accum
      MED med_phases_prep_wav_avg
      MED -> ATM :remapMethod=redist
      MED -> ICE :remapMethod=redist
      MED -> WAV :remapMethod=redist
      ATM phase1
      ATM -> CHM
      CHM
      CHM -> ATM
      ATM phase2
      ICE
      WAV
      ATM -> MED :remapMethod=redist
      MED med_phases_post_atm
      ICE -> MED :remapMethod=redist
      MED med_phases_post_ice
      WAV -> MED :remapMethod=redist
      MED med_phases_post_wav
      MED med_phases_ocnalb_run
      MED med_phases_prep_ocn_accum
    @
    OCN -> MED :remapMethod=redist
    MED med_phases_post_ocn
    MED med_phases_restart_write
  @
  ICE -> ICEIO :remapMethod=redist
  ICEIO
@
::
```

Future work:

- Work with ESMF team to develop a prototype NUOPC write component
  - Simple prototype that can satisfy the requirements
  - Challenges
    - Handshaking between model component and the write component as the write component does not know the output fields.
    - Simplify the run sequence
      - Random output time
      - Model component communicates with multiple write components
- Additional code needs to be provided to UFS model components
  - Communicate with authoritative component model managers
    - Agree upon the approach for asynchronized write component
    - Additional code updates on getting output fields information
- Performance testing

Thank you!