E3SM
Energy Exascale
Earth System Model

# SCREAM Update

Peter Caldwell, Andy Salinger, Luca Bertagna, Hassan Beydoun, Peter Bogenschutz, Andrew Bradley, Aaron Donahue, Chris Eldred, Jim Foucar, Chris Golaz, Oksana Guba, Ben Hillman, Rob Jacob, Jeff Johnson, Noel Keen, Jayesh Krishna, Wuyin Lin, Weiran Liu, Kyle Pressel, Balwinder Singh, Andrew Steyer, Mark Taylor, Chris Terai, Paul Ullrich, Danqing Wu, Xingqiu Yuan

Work from LLNL is performed under the auspices of the US DOE by LLNL under Contract DE-AC52-07NA27344.
UCRL:LLNL-PRES-796423

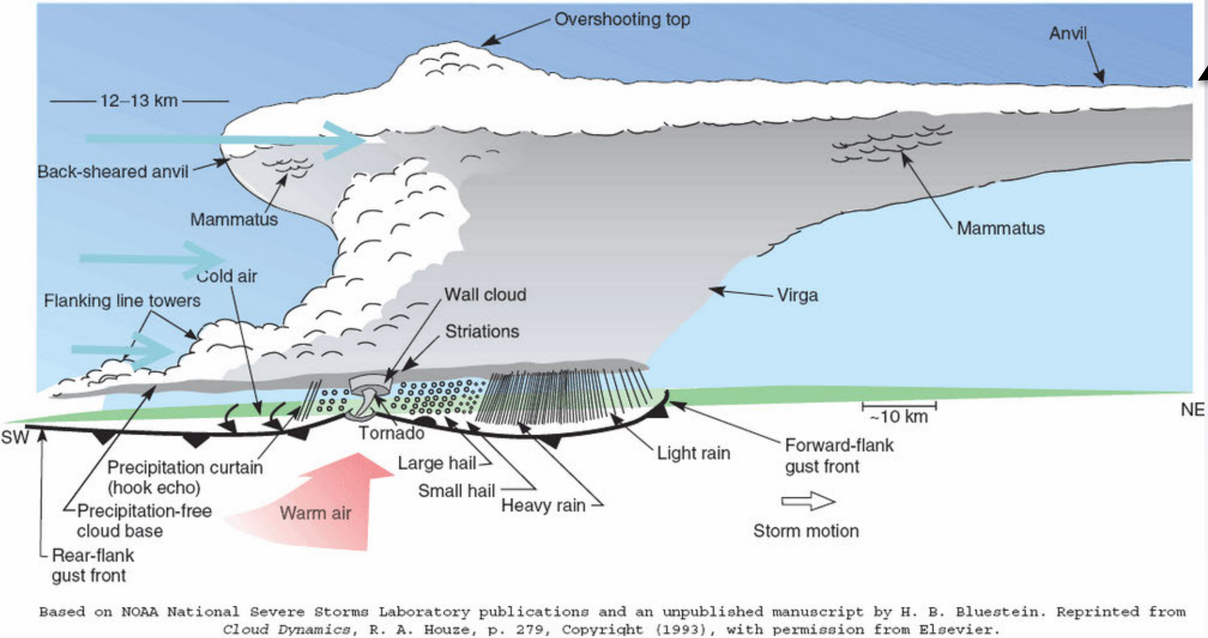
U.S. DEPARTMENT OF ENERGY

# 1. Background/Context

# Code Name:
# SCREAM - Simple Cloud-Resolving E3SM Atmosphere Model

1. Goal is to keep code as simple as possible
    a. You shouldn't trust results you don't understand physically…
    b. Simplicity makes clean rewrite (needed for performance) possible
    c. Resolving more makes complex parameterizations less important
2. Not quite cloud-resolving (yet!), but makes for a cool acronym
    a. Target dx = 3 km globally, 128 vertical layers with a top at 40 km
3. E3SM ocean and sea-ice already work at these scales
    a. Goal here is a *coupled* km-scale system, not just an atm model

# Why 3km Δx?




Schematic of a supercell thunderstorm from Houze' *Cloud Dynamics* (1993), extracted in color from inthecloudhead.blogspot.com .

- Many important climate processes occur on km scale
- Parameterizing these processes is uncertain/hard... resolving them increases prediction credibility
- Climate impacts will be felt on local scales… so we'd better be able to simulate them!

# Thesis: Climate Change Can be Understood from Short Runs

This thesis is critical to SCREAM because high-res requires short timesteps (for CFL stability), limiting simulation length

Reasonable because:

- Clouds are the main source of climate uncertainty
- Clouds respond rapidly to forcing change

  ⇒ Cleverly-designed short runs should tell us a lot

- Several clever short tests already exist
  - 5 yr Cess (prescribed SST increase) runs
  - 15 mo aerosol sensitivity tests nudged to observations

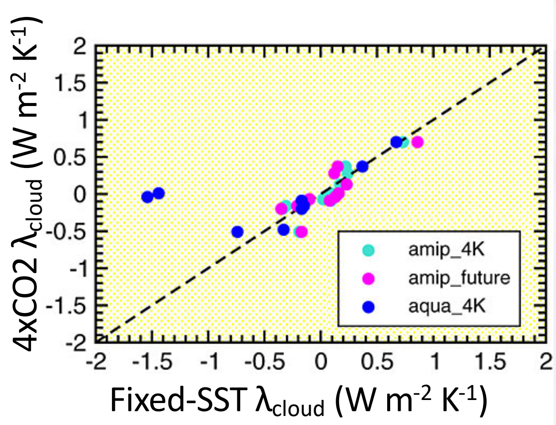


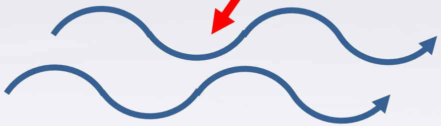

*Fig: Cloud feedback from full-complexity (y-axis) versus fixed SST simulations in CMIP5. Adapted from Ringer et al, (2014 GRL).*

# Overview of Processes

~~parameterized convection~~

**Turbulence** and **cloud formation** handled by Simplified Higher-Order Closure (**SHOC)**

Resolved-scale **fluid dynamics** treated by a non-hydrostatic Spectral Element (**SE**) approach



**Radiation** handled by externally-developed, GPU-ready **RRTMGP** package

**Aerosols** will be prescribed in initial implementation

**Microphysical processes** handled by Predicted Particle Properties (**P3)** scheme

# Other Efforts: DYAMOND

- Many other groups are working on global storm-resolving models (GSRMs)
- 9 models (on right) participated last year in an intercomparison of 40 day simulations with dx<5 km globally ([www.esiwace.eu/services/dyamond](www.esiwace.eu/services/dyamond))
- SCREAM is playing catch up, but will bring some unique elements…


*Fig: Snapshot of cloud fields from Aug 4, 2016 from GSRMs and Himawari satellite*

# E3SM Novelties:

1. SHOC is a more sophisticated cloud/turbulence parameterization (see fig)
2. Obsession with testing will yield a more trustworthy model
3. SCREAM will be written in C++/Kokkos (for GPUs)
4. GPU support, fast dycore, streamlined code, and access to exascale machines ⇒ should be able to do longer/higher-res runs
5. Access to eddying ocean/ice enables coupled runs

Cloud liquid water mixing ratio

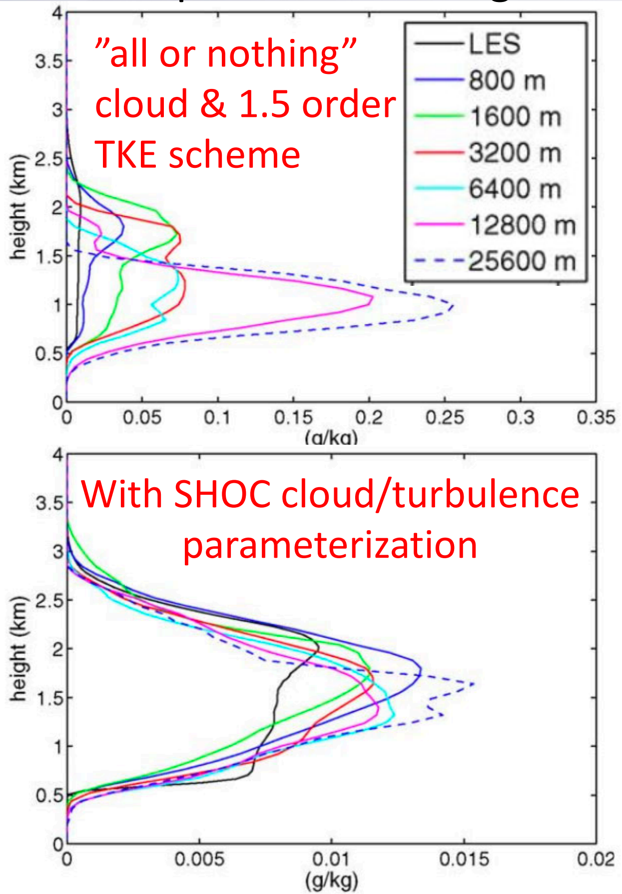
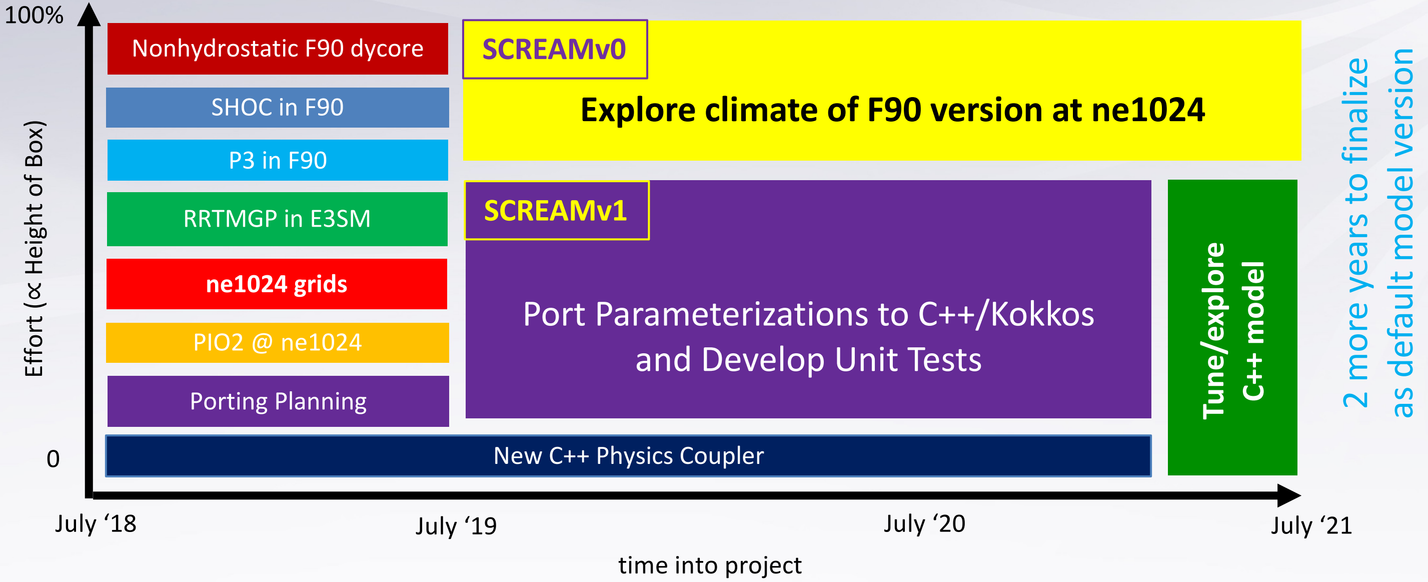



*Fig: SAM CRM simulations of RICO shallow cumulus with various horizontal resolutions (Bogenschutz and Krueger 2013)*

# Revised Plan (Crude):

- 3 stages:
  1. develop schemes
  2. Explore F90
  3. Create C++
- Stages 2 and 3 occur in parallel
- Behind schedule from original (very ambitious) plan but making great progress




* individual task timings above are totally off – goal is to provide broad overview

# 2. SCREAMv0

# Status: SCREAMv0 (F90 version)

- We can run at ne1024!
  - Runs in real time: 1 simulated day per day 😦 (but not tuned at all + using few nodes)
  - Using Scorpio/PIO2 for output, but not Adios (=each output node writes its own file)
  - Simulated world looks like earth (mostly)

- Near-term plan:
  - Participate in DYAMOND Phase 2
    - Provides credibility and access  to knowledge from other groups
    - Input data was just made available Tuesday. Runs due end of 2020(?)

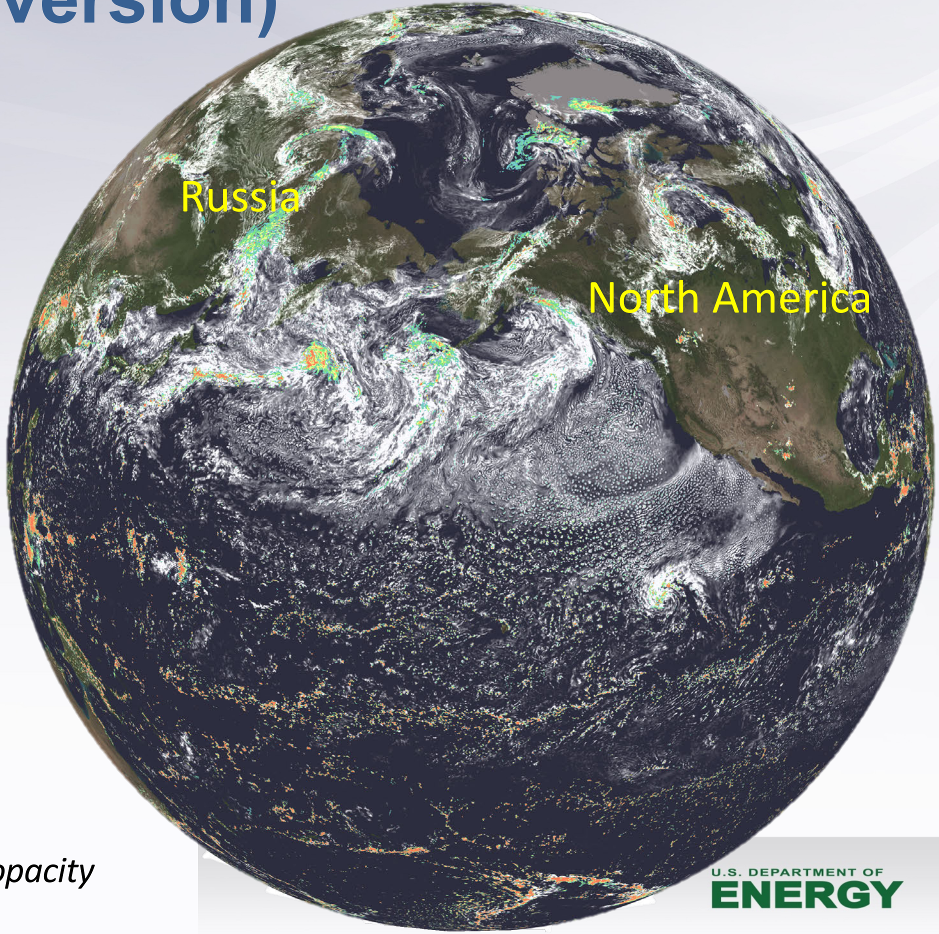



*Fig: Snapshot of precipitation (color) and liquid water path (opacity with opaque white = 200 g m$^{-2}$) after 2.5 simulated days.*

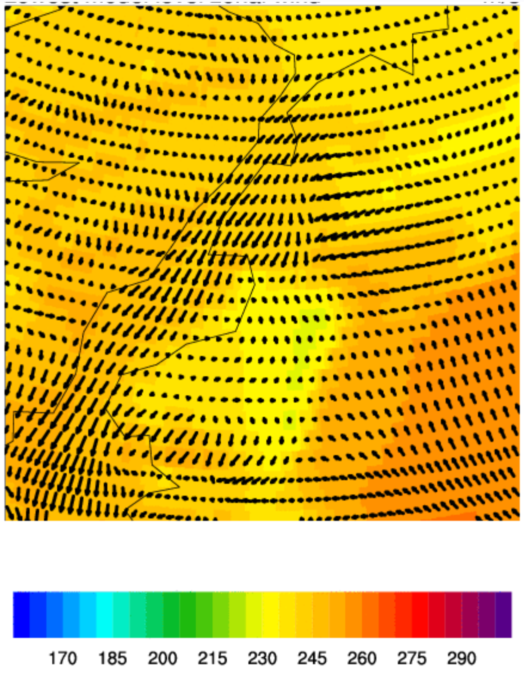
# "Minor" Issues with SCREAMv0

1. "Tlay" bug: brief periods of < -200 C surface temperature
    a. Gets worse at higher resolution
    b. Seems to be related to dycore (and/or its coupling with physics)

2. F90 threading problem in P3(?) restarts

3. Still haven't tuned at all for performance or skill

**ne256 Cold Event**




*ridiculously*
*V*
*Movie: Near-surface winds (arrows) and T (colors) during cold event on W coast of Greenland on Jan 27th @ne256.*

# Short Forecast Runs

Weather forecasts will be a major evaluation tool because climate runs are too expensive. There are 3 ways to generate initial conditions for forecasts:

1. CAPT (Ma + Klein): the original
2. Betacast (Zarzycki): meant to be easier to use for the univ community
3. HIPPO (Hannah): newest/simplest

We are intercomparing methods to choose the best approach for SCREAM and are encouraging folks to merge their methods

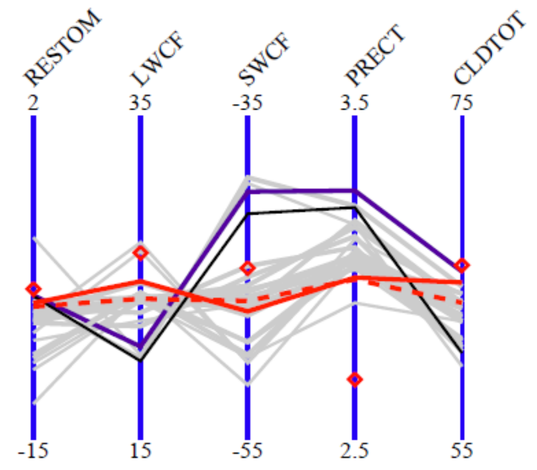
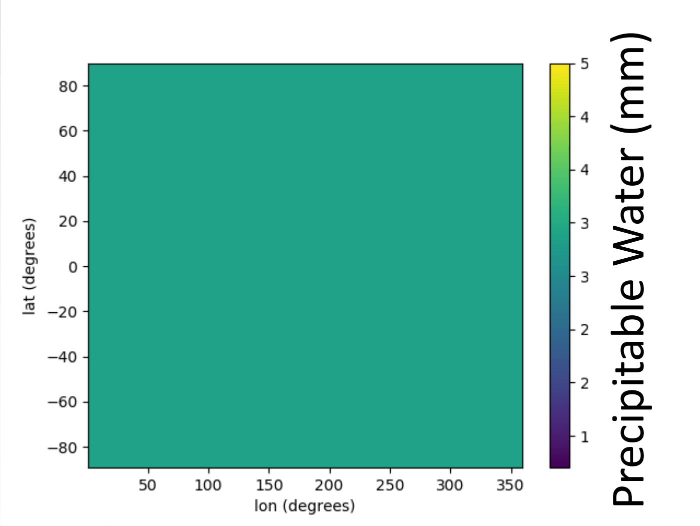


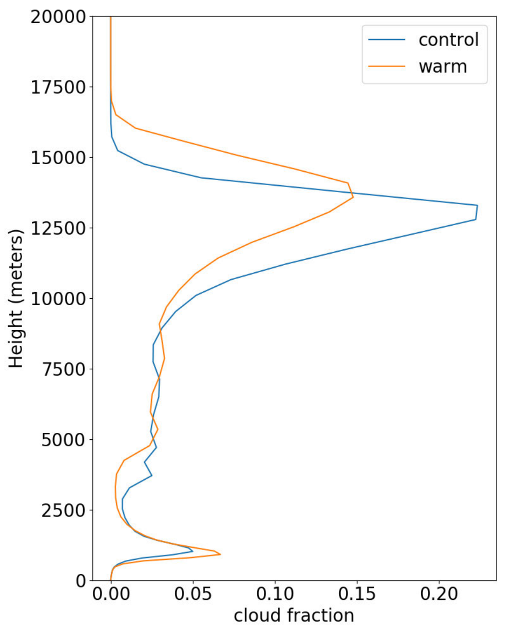

*Fig: Key global-average metrics for June-Aug simulations. Diamonds indicate observed values, gray lines are 24-48 hr forecast runs, and black lines are free-running simulations. Red lines indicate forecast (dashed) and free-running (solid) simulations for one particular tuning. Courtesy Wuyin Lin.*

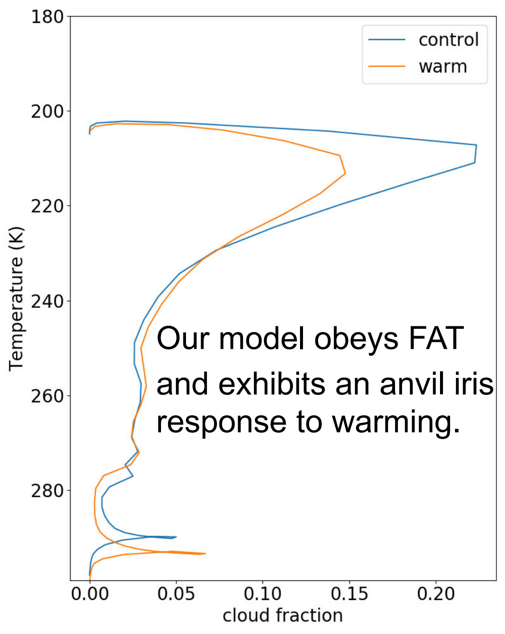
# SCREAMv0 Radiative Convective Equilibrium (RCE)

Like other models, SCREAM Self-aggregates in RCE

Running varying SST simulations to look at cloud response to warming as well as model physical soundness in an idealized setup.



# SCREAMv1

# Coding

SCREAM will be rewritten from scratch in C++:

- Allows use of Kokkos library, which abstracts on-node parallelism
  - single code runs efficiently on CPUs, GPUs, etc
  - Unlocks more parallelism…
  - But results in more complex code

**Original F90**

```fortran
kloop_sedi_c2: do k = k_qxtop,k_qxbot,-kdir

   qc_notsmall_c2: if (qc_incld(k)>qsmall) then
      !-- compute Vq, Vn
      call get_cloud_dsd2(qc_incld(k),nc_incld(k),mu_c(k),rho(k),nu,dnu,   &
           lamc(k),tmp1,tmp2,lcldm(k))

      nc(k) = nc_incld(k)*lcldm(k)
      dum = 1._rtype / bfb_pow(lamc(k), bcn)
      V_qc(k) = acn(k)*bfb_gamma(4._rtype+bcn+mu_c(k))*dum/(bfb_gamma(mu_c(k)+4._rtype))
      V_nc(k) = acn(k)*bfb_gamma(1._rtype+bcn+mu_c(k))*dum/(bfb_gamma(mu_c(k)+1._rtype))

   endif qc_notsmall_c2
   Co_max = max(Co_max, V_qc(k)*dt_left*inv_dzq(k))

enddo kloop_sedi_c2
```

**Ported to C++/Kokkos**

```cpp
Kokkos::parallel_reduce(
  Kokkos::TeamThreadRange(team, kmax-kmin+1), [&] (int pk_, Scalar& lmax) {
    const int pk = kmin + pk_;
    const auto range_pack = scream::pack::range<IntSmallPack>(pk*Spack::n);
    const auto range_mask = range_pack >= kmin_scalar && range_pack <= kmax_scalar;
    const auto qc_gt_small = range_mask && qc_incld(pk) > qsmall;
    if (qc_gt_small.any()) {
      // compute Vq, Vn
      Spack nu, cdist, cdist1, dum;
      get_cloud_dsd2<false>(qc_gt_small, qc_incld(pk), nc_incld(pk), mu_c(pk), rho(pk), nu, dnu, lamc(pk), cdist
      nc(pk).set(qc_gt_small, nc_incld(pk)*lcldm(pk));
      dum = 1 / (pack::pow(lamc(pk), bcn));
      V_qc(pk).set(qc_gt_small, acn(pk)*pack::tgamma(4 + bcn + mu_c(pk)) * dum / (pack::tgamma(mu_c(pk)+4)));
      if (log_predictNc) {
        V_nc(pk).set(qc_gt_small, acn(pk)*pack::tgamma(1 + bcn + mu_c(pk)) * dum / (pack::tgamma(mu_c(pk)+1)));
      }

      const auto Co_max_local = max(qc_gt_small, -1,
                                    V_qc(pk) * dt_left * inv_dzq(pk));
      if (Co_max_local > lmax)
        lmax = Co_max_local;
    }
}, Kokkos::Max<Scalar>(Co_max));
team.team_barrier();
```

# Dycore Status

- Done porting of C++ version of nonhydrostatic (NH) dycore
  - Currently being used for Gordon-Bell submission (see fig)
- We get almost 1 SYPD using all of Summit (see fig).
  - Does not include semi-Lagrangian advection ⇒ further speed up
- Dynamics dt will be 9.375 sec and advection+phys/dyn coupling dt will be 75 sec **for real-world runs**

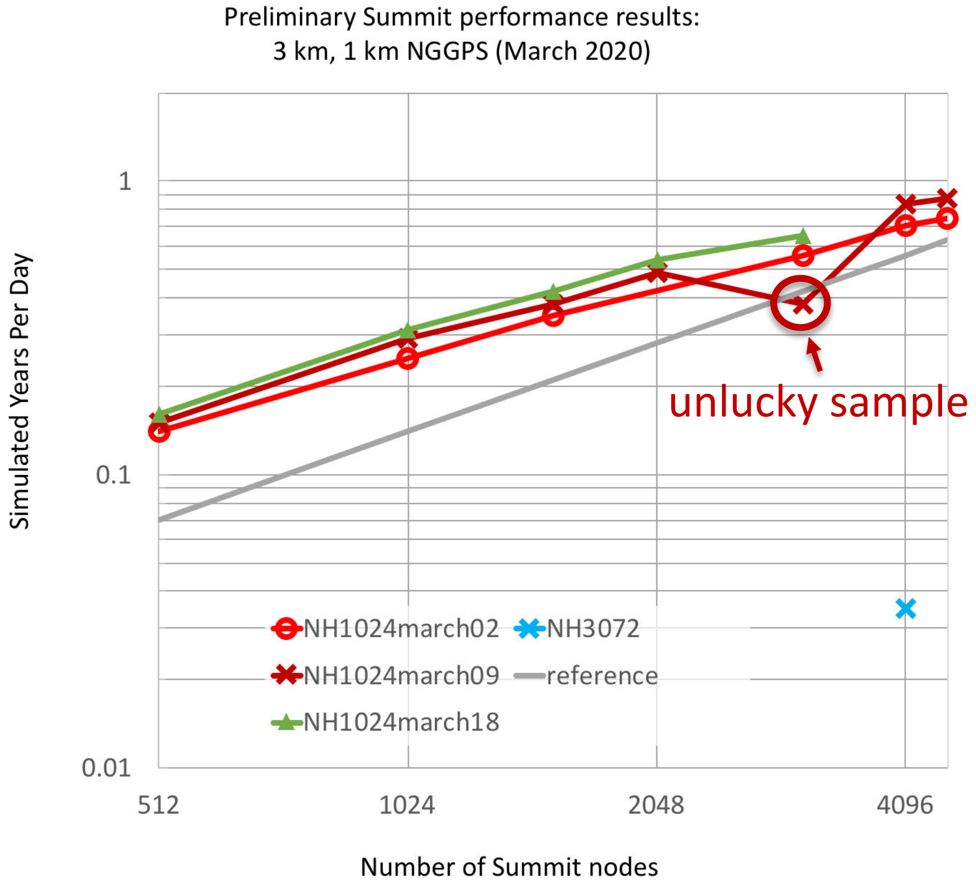



*Fig: Nonhydrostatic C++ dycore-only NGGPS timings at ne1024 and ne3072 (blue x). Different colored lines are for earlier or later versions of code.*

# Physics Status

- P3 port is wrapping up
  - Served as great training: 6 people learned how to port!
- SHOC port is just starting
- Beginning prep work on RRTMGP interface
- Prescribed aerosol implementation still needs thought

Porting Steps:

1. Create standalone driver to enable fast BFB testing
2. Clean up code/break it into small subroutines
3. Make table of dependencies to decide order of subroutine porting
4. Port subroutines, using BFB tests to avoid bugs

# Process Coupling Status

1. Atmospheric processes are all instances of the **atm_process class**. Having all processes behave the same:
   a. makes adding/reordering/parallelizing processes easy
   b. improves code-reusability
2. Processes are broken into:
   a. SCREAM-specific **interface layers**
   b. Model-agnostic **process implementations** which make:
      i. our code easy to share with other models
      ii. standalone process simulations straightforward

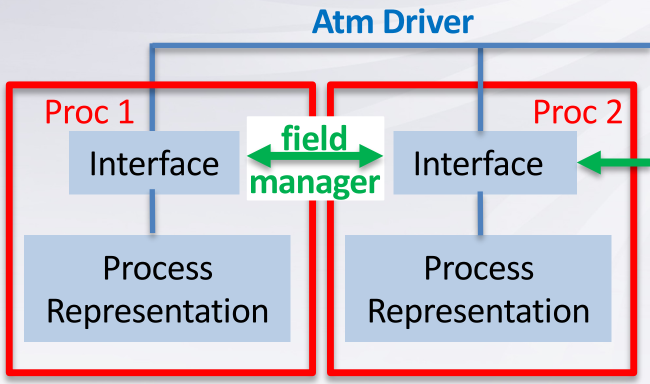



*Fig: SCREAM coupler structure*

3. Processes communicate entirely through a **field manager** which provides interface layers pointers to requested variables
   a. the field manager is like a modernized pbuf, but will be used for *all* variables

# Testing

- Goal is *property tests* (checking code is behaving physically) in addition to regression tests (checking answers haven't changed)
- Convergence in $\Delta t$, $\Delta z$, and $\Delta x$ is a good property test. Applying to SHOC (see fig) revealed problems with:
  - the Bretherton+Park (2009) shear production boundary condition
  - the Blackadar (1984) turbulent length scale near the surface
- Encapsulation of parameterizations makes unit tests easy
- Finding time to write property tests is hard

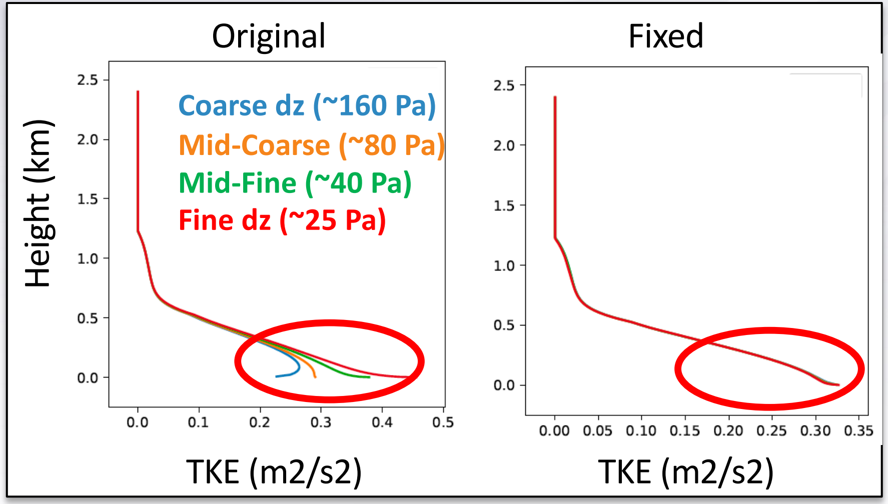



*Fig: SHOC standalone simulations running the BOMEX test case (trade wind Cumulus) for 6 hrs with a variety of vertical resolutions*

# Summary

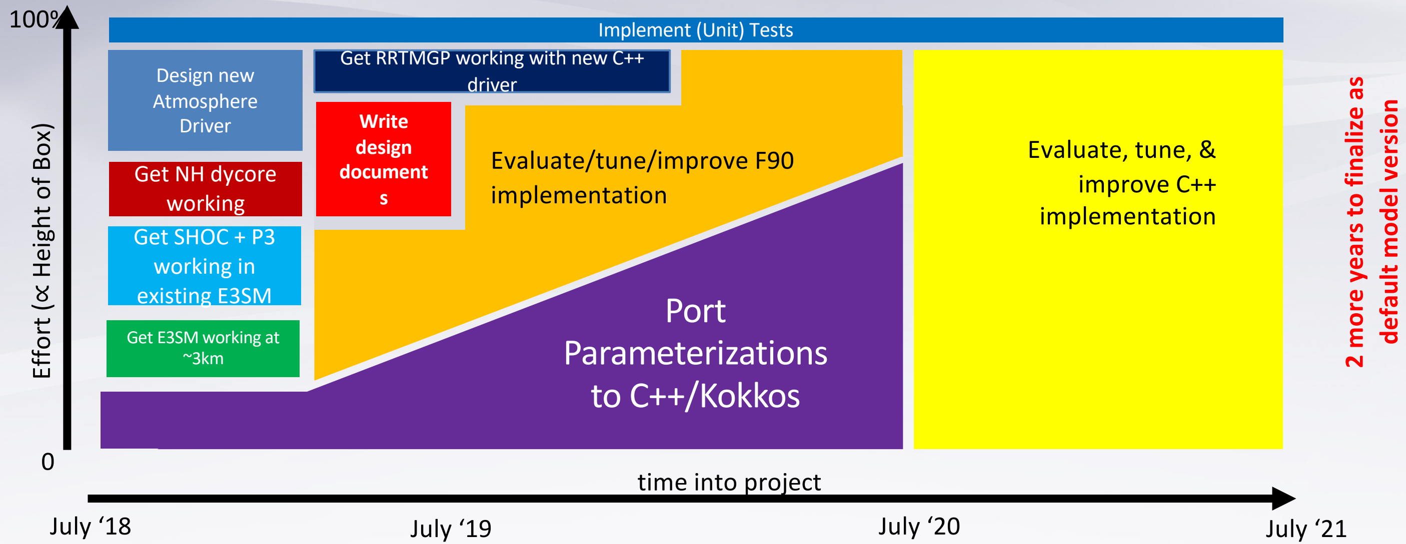
1. SCREAM is making great progress on many fronts
2. SCREAMv0 should be available for science soon
3. The move to C++/Kokkos is going well
4. Testing has caught many bugs but would benefit from even more attention
5. Exploratory science involving other components will make sense within a year

# Extra Slides

# Why You Should Care (Connection to Others):

- ALL: Are C++ or other design elements good for you too?
- ATM: Cutting edge/probably your future model
- LND: All runs includes $1/8^0$ land simulation– lots of science/improvement opportunities!
- OCN/ICE: ocn/atm/ice coupling at 3 km would be fascinating and good training data for coarser resolutions
- PERF: Lots of interesting opportunities/implications of moving to Kokkos
- INFRASTRUCTURE: Running and evaluating this model will be a challenge!
- BGC: We will start to resolve urban scales – connection to human systems?

# Original Plan:




Strategy:

1. Get initial version working quickly for testing and to act as a template
2. Plan for a full year of fixing performance and skill