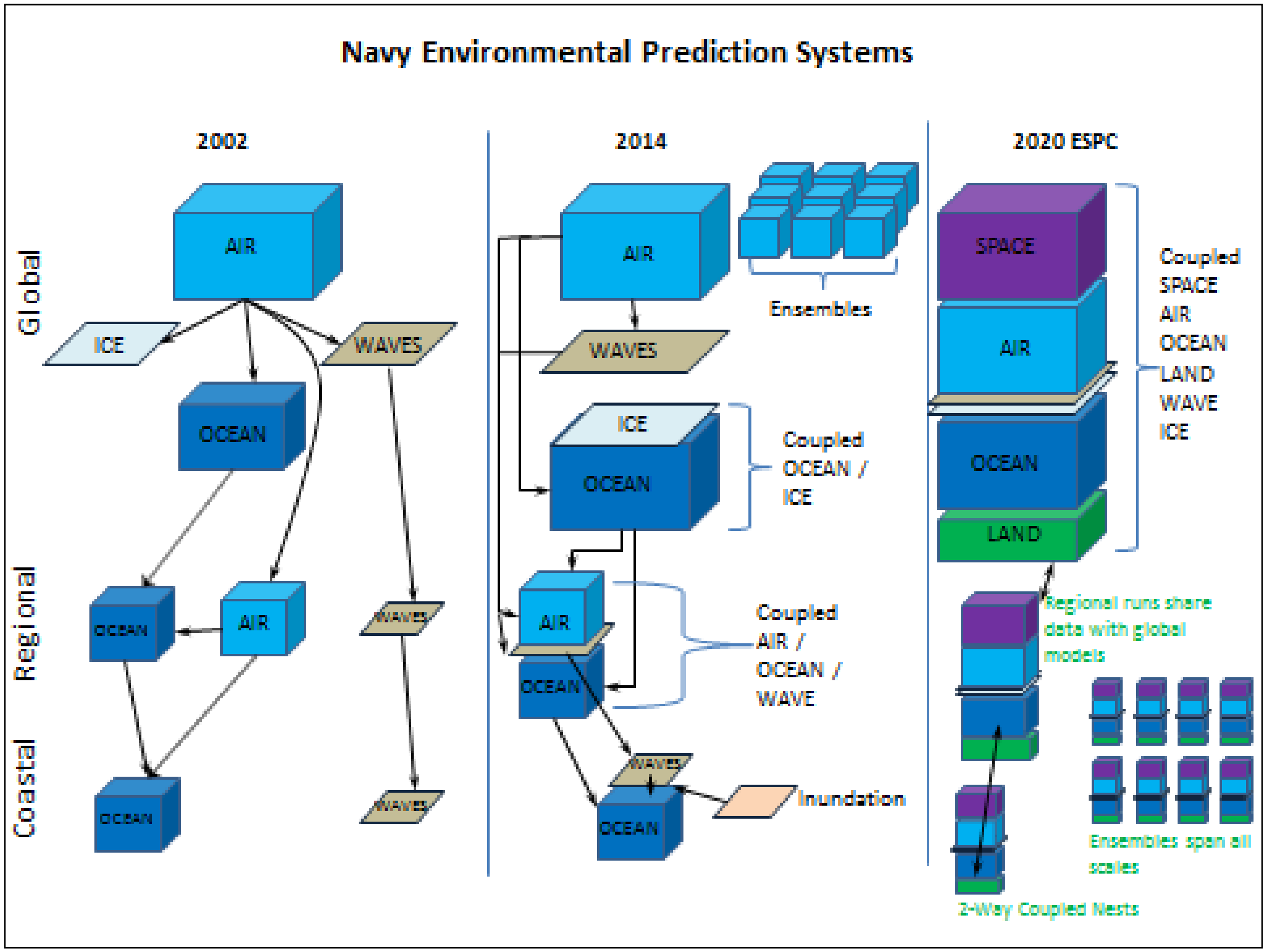
HYCOM and Navy ESPC Future High Performance Computing Needs

Alan J. Wallcraft

COAPS Short Seminar

November 6, 2017

Forecasting Architectural Trends



NAVY OPERATIONAL GLOBAL OCEAN PREDICTION

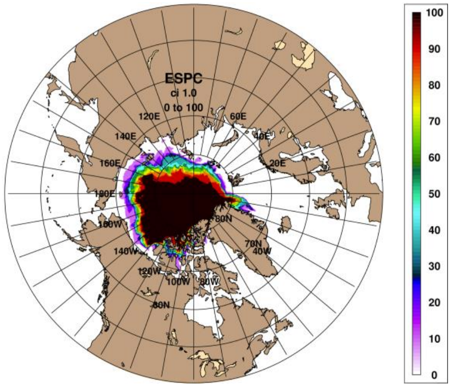
- **Trend is higher resolution and coupling to other environmental components**
  - **Global Ocean Forecasting System (GOFS)**
  - **Navy Earth System Prediction System (ESPC)**
- **GOFS 3.0: 1/12° 32 layer HYCOM (ocean)**
  - **Operational 20 March 2013**
    - **http://hycom.org/ocean-prediction for images and movies**
    - **http://hycom.org/dataserver/glb-analysis for model fields**
- **GOFS 3.1: 1/12° 41 layer HYCOM/CICEv4 (ocean/sea ice)**
  - **Transitioned from NRL to NAVO FY17Q2**
    - **Will be available to the public via http://hycom.org**
- **GOFS 3.5: 1/25° 41 layer HYCOM/CICEv5/tides**
  - **Planned transition from NRL to NAVO in FY18**
    - **Model fields will not be at http://hycom.org**
- **Navy ESPC 1.0: HYCOM+CICEv5+NAVGEM+WW3 (ocean/sea ice/atmosphere/waves)**
  - **Initial Operational Capability (IOC) in 2018; Final OC (FOC) 2022**
- **Once it is formally operational (2022?), ESPC replaces GOFS**

Earth System Prediction Capability (ESPC)

U.S. NAVAL RESEARCH LABORATORY

- National, multi-agency collaborative effort to leverage resources to develop the next generation whole earth prediction system at timescales beyond synoptic weather forecasts
- Includes components:
  atmosphere/ocean/ice/waves/land/aerosol
- Runs in fully coupled mode including an ensemble prediction capability
- Provide guidance in forecasting:
  - Arctic sea ice extent and seasonal ice free data
  - Extreme weather events
  - Extend lead-time for tropical cyclone prediction

ESPC seasonal ice forecast



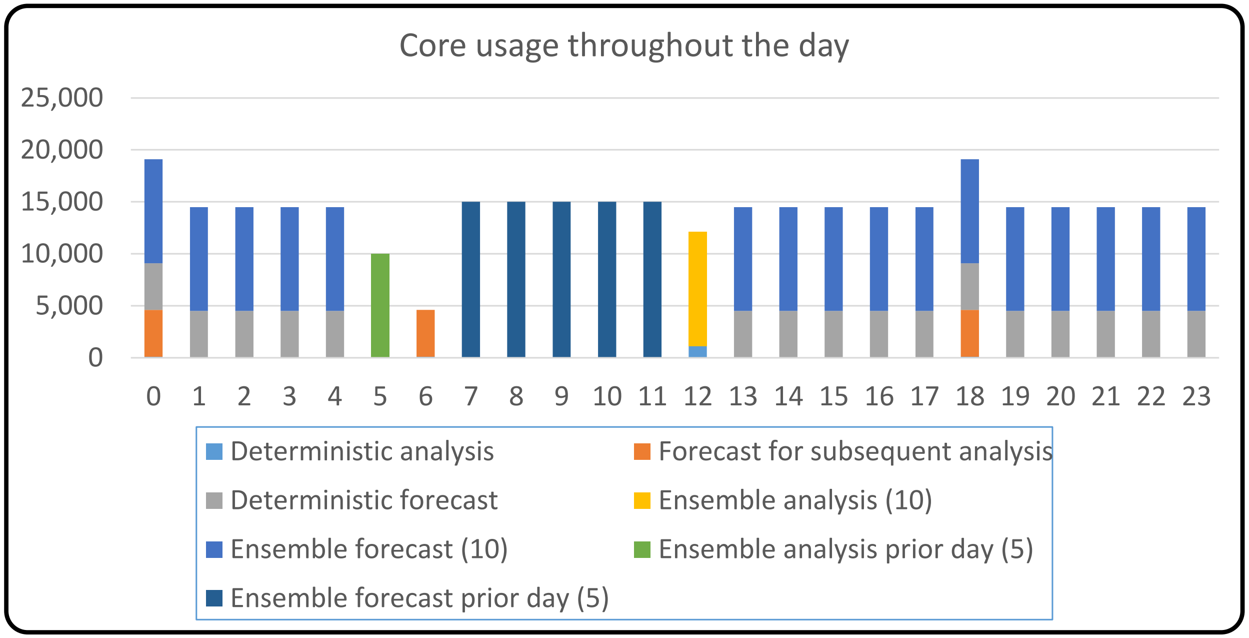
Navy's ESPC first generation system is scheduled to be running in real-time by 2018.

NAVY ESPC

- **Most ESPC systems focus on the atmosphere**
  - **Relatively low resolution ocean and sea ice**
  - **Might not includes waves**
- **Navy ESPC focuses on the entire Earth system**
  - **Resolves ocean fronts and eddies**
    - **1/12° HYCOM is 80% of ESPC cpu requirements**
  - **1/4° HYCOM would reduce total ESPC cost by 10x**
- **Major components from existing Navy CWO products**
  - **Lots of in-house experience with these components**
  - **Not necessarily designed for long forecasts**
    - **HYCOM and CICE have been used in multi-year simulations with a prescribed atmosphere**
    - **NAVGEM required significant re-tuning**
    - **Tuning of the coupled system is on going**
  - **Still much less effort than adopting new climate-focused components**
    - **Already work well with the in-place data assimilation systems**
    - **Tuning, testing and verification of new components for a forecast system requires time and many resources**

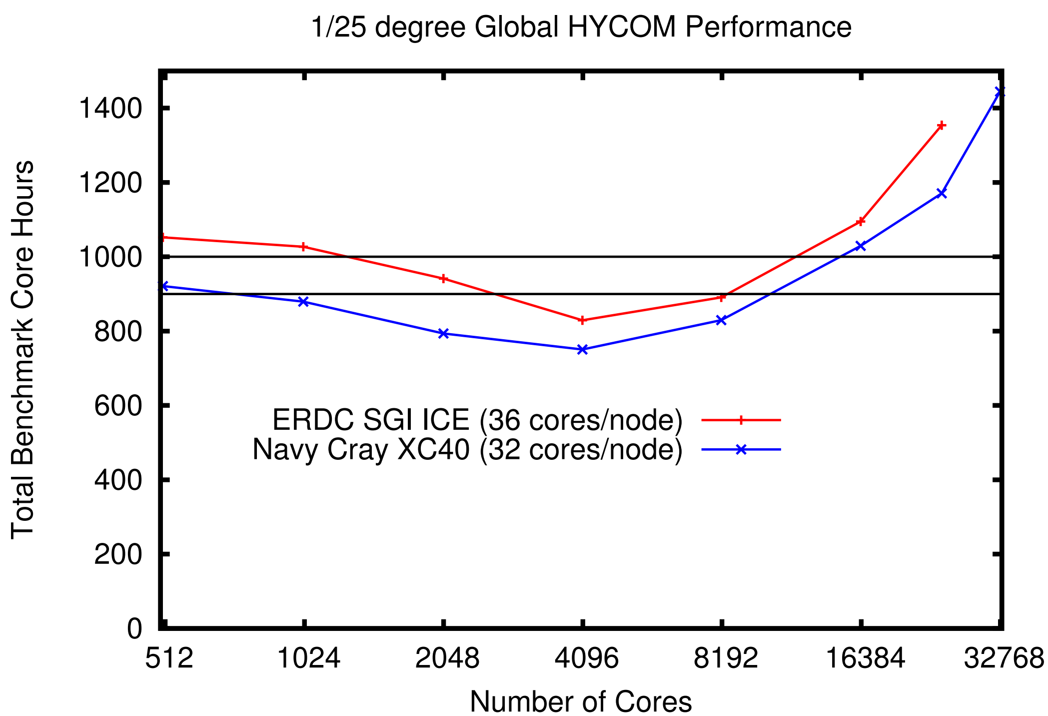
2018 IOC Configuration

Deterministic 16 day forecast: 1/25° HYCOM, 1/25° CICE, T681L80 NAVGEM, 1/8° WW3
Ensemble (15) 30 day forecast: 1/12° HYCOM, 1/12° CICE, T359L60 NAVGEM, 1/4° WW3
Total output per day ~221 TB
Hourly global 3D information



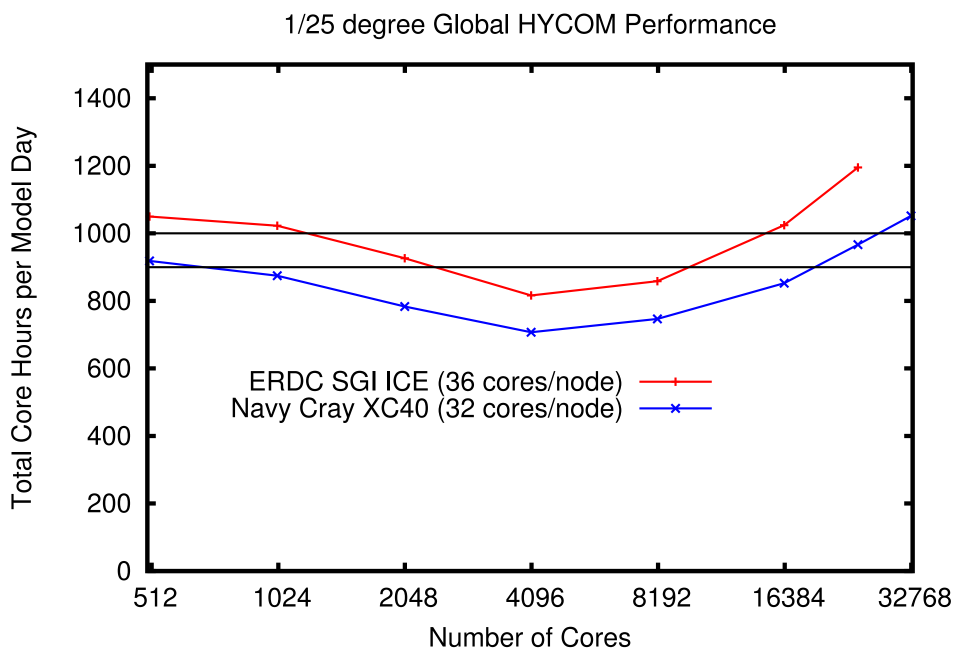
HYCOM HPCMP TI-16 BENCHMARK PERFORMANCE

- **Wall-clock time (start to end) for a 1 model day 1/25° fully global run**
- **Grid size 9000 x 6595 x 32**
- **Run includes typical I/O and data sampling**
- **Compiler options set for bit for bit results across any num. cpus**
  - **Probably not required for TI-16**

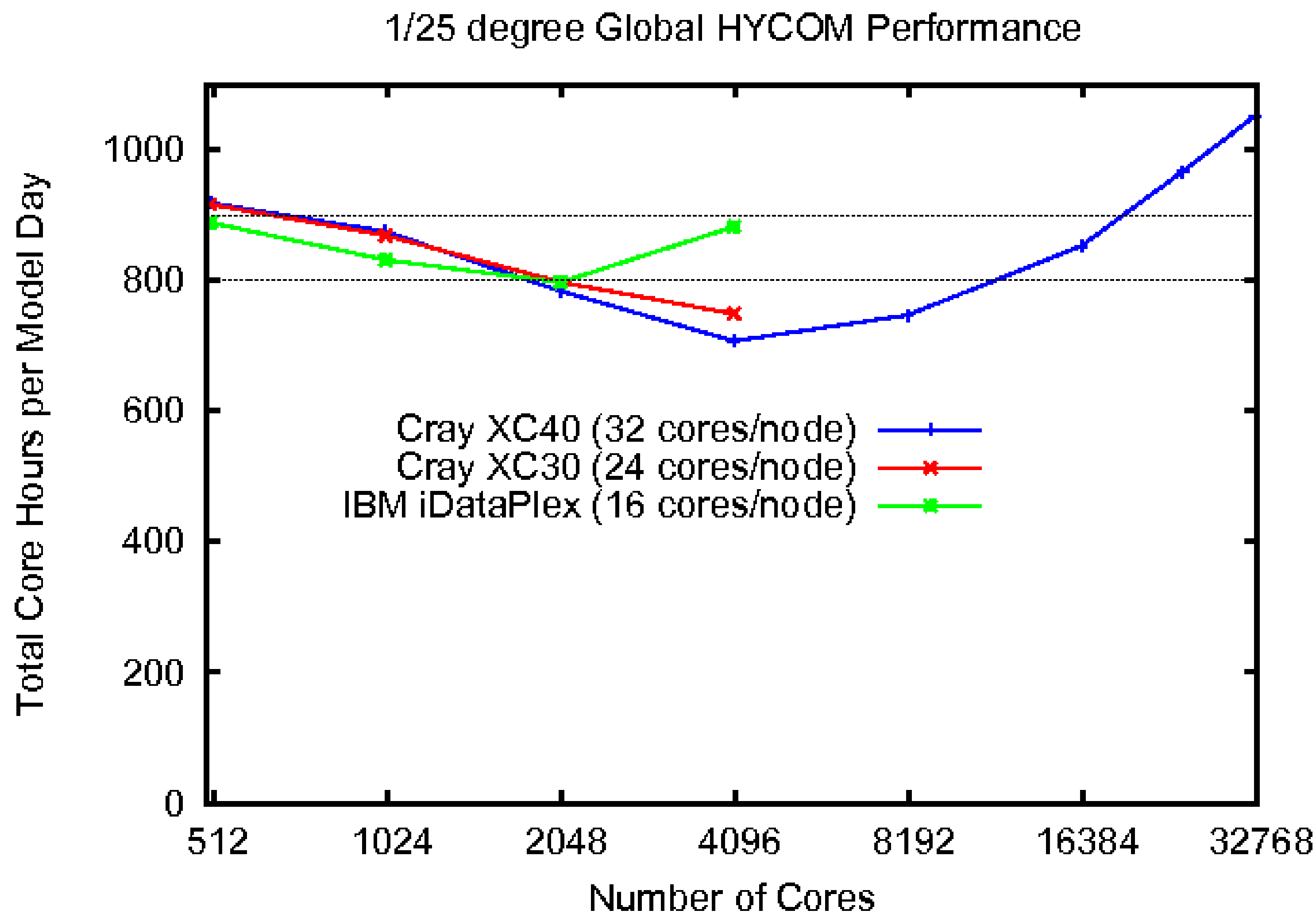


HYCOM TI-16 BENCHMARK PERFORMANCE PER MODEL DAY

- **Data assimilative runs are about 1 model day, but forecasts are currently 7 model days and will soon be 16 to 30 model days**
- **Wall-clock time for 1 model day 1/25° fully global case**
  - **Excludes wall time before the 1st time step**
  - **Parallel (MPI-2) I/O is the primary limit on scalability**
- **Same performance on 1K and 16K cores**



HYCOM PAST PERFORMANCE FOR GLOBAL 1/25°



- **Three generations of Intel Xeon with little difference in performance**
  - **Dual socket nodes: Sandybridge, Ivybridge, Haswell**
  - **Moore's Law giving us more cores per socket**

BIT-FOR-BIT MULTI-CPU REPRODUCIBILITY

- **Repeating a single processor run:**
  - **Produces identical results**
- **Repeating a multi-processor run:**
  - **Produces different results, using either OpenMP or MPI**
    - **e.g. fastest global sum is non-reproducible**
  - **Unless programmer explicitly avoids non-reproducible operations**
- **Require reproducibility on any number of processors**
  - **Test a compiler/system setup once, rather than for every core count**
- **Can't use the highest level of compiler optimization**
  - **ifort -fp-model precise -no-fma**
  - **fp-model precise because vector and scalar operations have different rounding, so the start and end of loop extents can't be scalar if the middle is vector**
  - **fused multiply-add is new with AVX2, it has different rounding and so must be used for all elements in a loop or none**
- **The Intel compiler is not providing the fastest possible reproducible results**
  - **In some cases this can be worked around with extra coding**
  - **This should not be necessary**

FUTURE SYSTEM UNCERTAINTY

- **Operational products take 5+ years to develop and have a 5+ year lifetime**
  - **Must target HPC systems 5-10 years in the future**
- **For more than a decade clusters of "fat" commodity core nodes have been the HPC systems of choice**
  - **Initially with cores from Intel, AMD and IBM (POWER)**
    - **Compiler differences and MPI library differences**
  - **More recently standardizing on Intel Xeon and Intel Fortran**
    - **IBM POWER still a viable option**
- **This has greatly simplified designing future operational products**
- **The HPC landscape is changing, making looking ahead much harder**
  - **Fat nodes may still be viable, with 48 or 64 (or 96) cores per node**
    - **Will Intel build them, what about power and memory performance?**
  - **Attached processors have higher peak performance across several metrics**
    - **Ocean models operate well away from this peak**
  - **Many-core systems becoming available**
    - **Hostless Intel Phi and perhaps others (ARM-based)**
    - **Simpler cores, but optimized for HPC**
    - **Is the HPC market large enough to sustain development?**

OCEAN MODELS ON ATTACHED PROCESSORS

- The low computational intensity of ocean models has been a issue on attached processors
- Cost of repeatedly moving arrays from system (host) memory to attached memory is prohibitive
- Only viable approach:
  - Copy all model arrays to attached memory
  - Run MPI across attached processors (without involving the host)
  - Use the host only for start up and I/O
    - I/O includes error reporting, which may require re-factoring the error handler
- This means that "incremental" approaches to porting won't work
  - Can't do one subroutine at a time
- The attached processor must have enough memory to hold all arrays
  - 1/25° HYCOM requires 850GB of memory plus tiling overhead
- Still must face the low computational intensity bottleneck
  - May not get good performance without major code re-factoring

OCEAN MODELS ON FUTURE SYSTEMS

- **The memory and programming limitations of attached processors are being reduced over time**
  - **Make host memory more accessible and increase size of "fast" memory**
  - **Host-less "attached" processors, with "fast" memory treated as a cache**
- **Host-less approach is also "many simpler cores" vs "fewer faster cores"**
  - **Currently Intel Knights Landing single socket node with 72 cores per socket vs Intel Xeon dual socket nodes with 18 (say) cores per socket**
  - **Knights Landing has enhanced vector operations (i.e. optimized for HPC) but may require more use of Hyper-Threading for good performance**
  - **72 vs 36 cores per node. Which is a) faster per node, b) faster per watt, or c) faster per dollar?**
- **In the future ARM server chips with vector extensions will join the "many-cores" class**
- **In general, ensembles of ocean models scale well (favors more cores) but may need re-factoring to take advantage of vector hardware**
  - **Knights Landing may need Hyper-Threading for maximum performance**
    - **Increase the number of MPI tasks, or use MPI and OpenMP**

SUMMARY

- **The HPC landscape has been very stable, but its future is less clear**
- **There are some things we can do that are future agnostic**
  - **Improved vectorization**
  - **Memory hierarchy optimization**
    - **At a minimum gets us better cache use**
- **We need more targeted compiler optimizations**
  - **Better support for bit for bit reproducibility**
  - **Perhaps enhanced in-lining**
  - **Only the compiler vendors can provide this**
- **If attached processors become dominant, then CWO applications will be at a severe disadvantage**
  - **Is this like shared memory vector vs distributed memory MPI?**
    - **HPCMP bought a few shared memory systems for a while, but codes that did not switch to MPI faded away**
- **The HPC-optimized many-core approach (e.g. Intel Phi) is as yet untested on ocean models, but may work well for CWO**