High performance computing enhancements to WRFV3.5

Tianhe-2, #1 Top 500 list, June 2013

John Michalakes

Computational Sciences Center
National Renewable Energy Laboratory

WRF Users Workshop

June 25, 2013

Contributors

- Intel Corp.
  - Michael Greenfield
  - Lawrence Meadows
  - Alexander Knyazev
  - Indraneil Gokhale
  - Ruchira Sasanka
- NCAR
  - WRF Developers Committee
- U. Wisconsin/SSEC
  - Jarno Mielikainen
  - Bormin Huang
- Alexandria University, Egypt
  - Ahmed Saeed et al.
- U. Colorado
  - Jeremy Siek
- NREL
  - Jim Albin

NUDT Tianhe-2 (China) and Intel Phi

- **#1,** June 2013 Top500 list
- 16,000 nodes, each with
  - 2 × hex-core Intel Xeon 2.2GHz (IvyBridge)
  - 3 × 57-core Xeon Phi 1.1 GHz
  - 30.65 PFLOPs (10^{18} FLOP s^{-1}) (Linpack)
- Previous #1 system: ORNL's Titan
  - 18,688 nodes each with AMD and NVIDIA Kepler 20x GPUs
  - 17.6 PFLOPs

Tianhe-2 System (above)
Intel Xeon Phi cards in Tianhe-2 (below)

From "Visit to the National University for Defense Technology Changsha, China." Jack Dongarra, University of Tennessee, and Oak Ridge National Laboratory. June 2013.
www.netlib.org/utk/people/JackDongarra/PAPERS/tianhe-2-dongarra-report.pdf

Many Integrated Core (MIC), Knights Corner

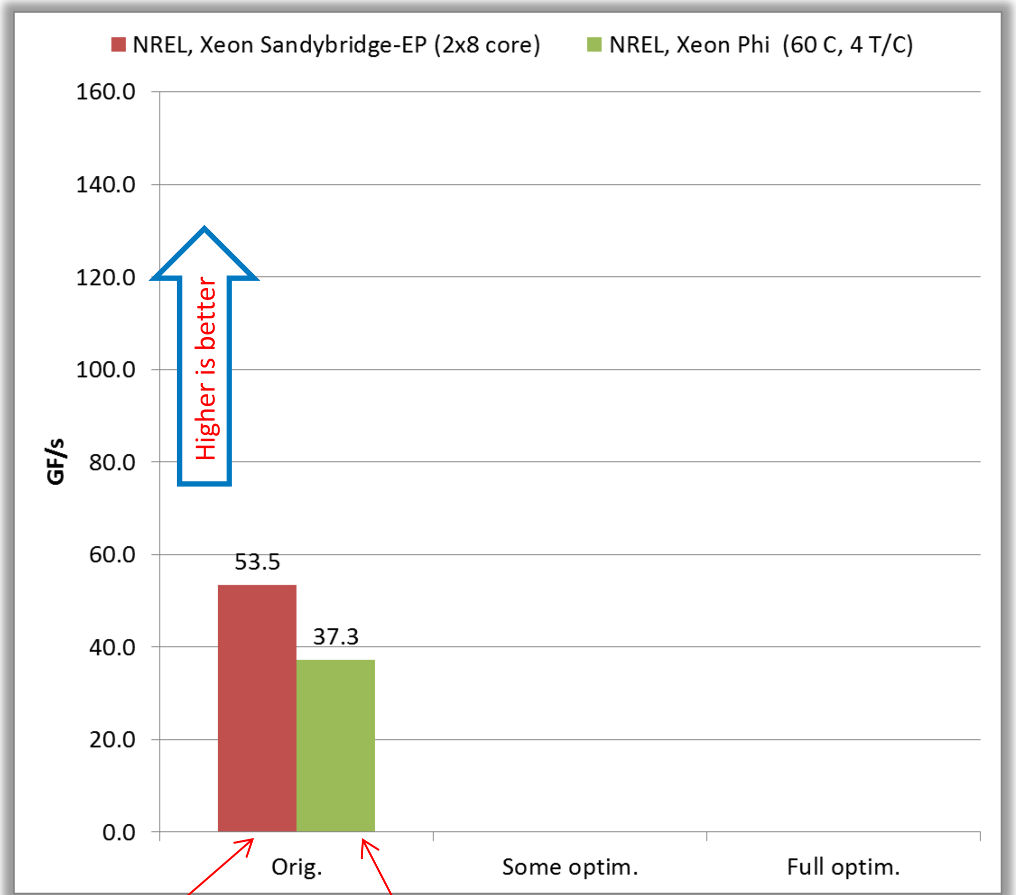
Teraflop "Cluster on a Chip"

- 61 Intel x86 CPUs (cores)
- Separate co-processor, but with its own network address
  - Supports login and shell
  - NFS-mounts host file systems
  - Runs as node in a larger MPI job
- Programmed like a cluster
  - Fortran/C/C++
  - MPI and OpenMP
  - WRF ported in 4 days (2011)
- Performance comes from concurrency (threads) and vectorization
- Porting straightforward, but optimization is still a task

Intel Phi's (KNC) in drawer of TH-2

WSM5 Microphysics

Optimization for Xeon Phi



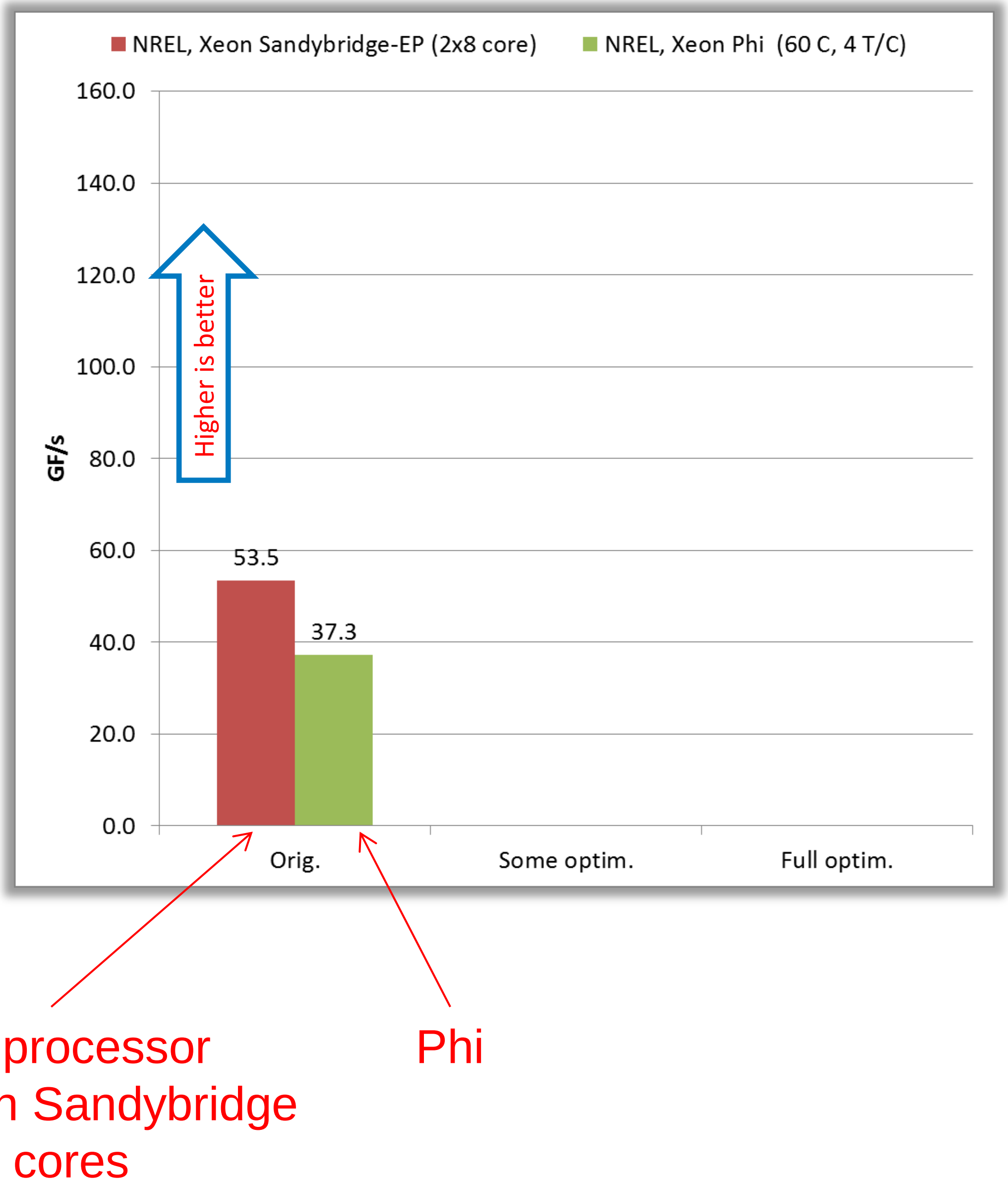
Host processor
Intel Xeon Sandybridge
8 cores

Phi

WSM5 Microphysics

Optimization for Xeon Phi

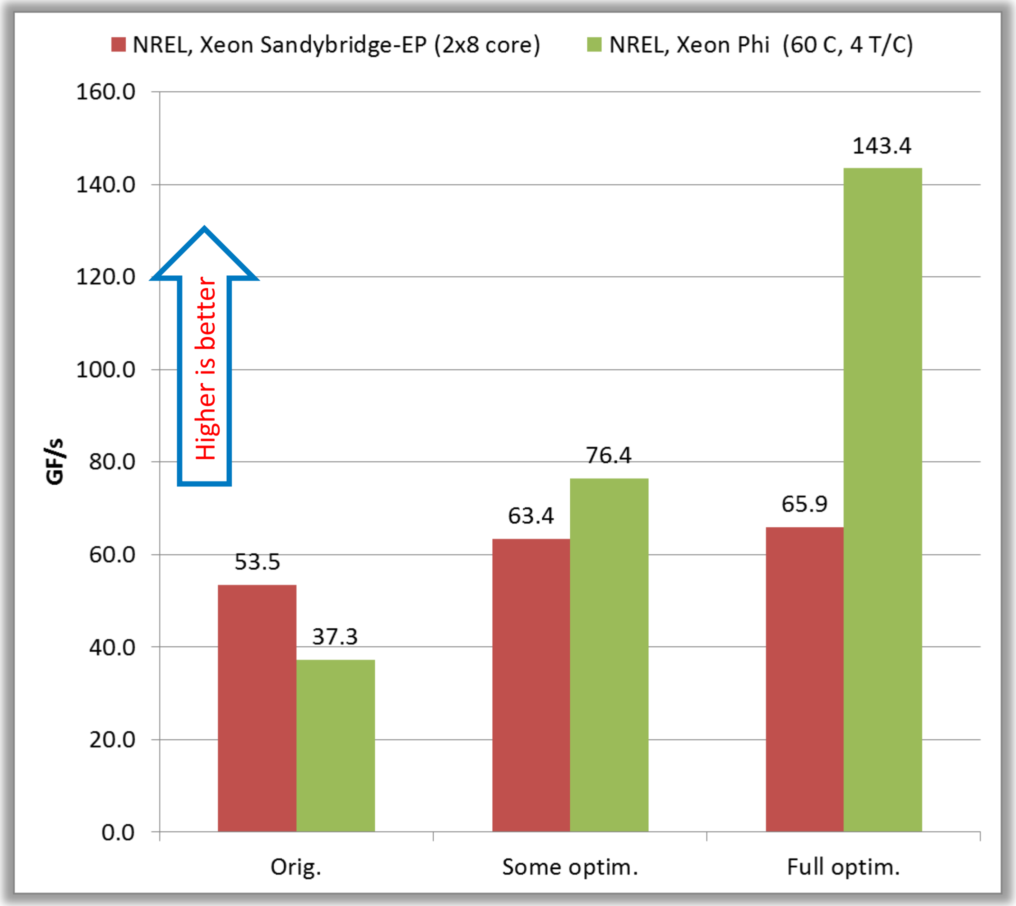
- Performance analysis showed
  - Inadequate vectorization from misalignment and loop peeling
  - Large memory footprint not fitting in L1 and L2 caches
  - Memory latency bound and not saturating memory bandwidth



WSM5 Microphysics

Optimization for Xeon Phi

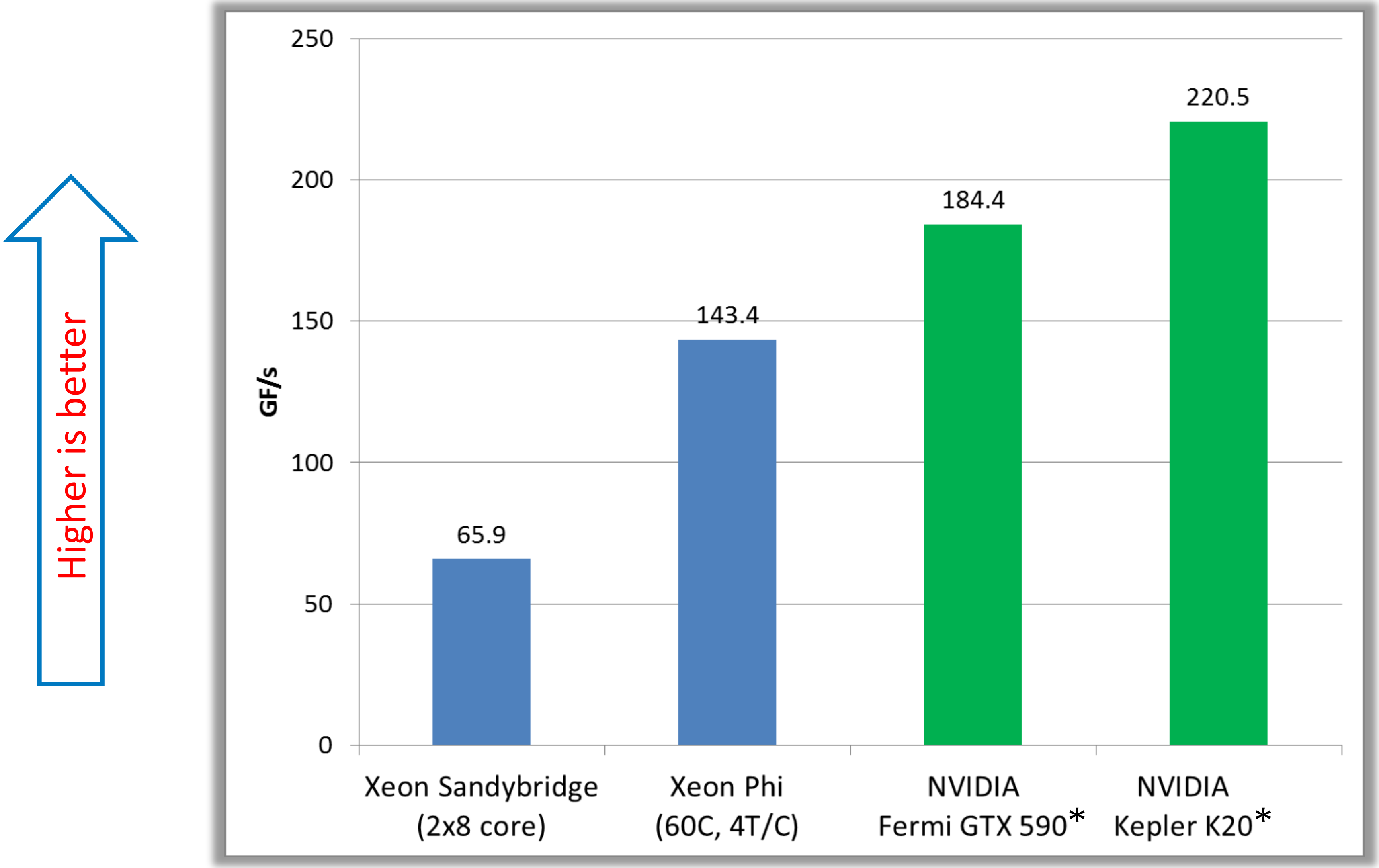
- Performance analysis showed
  - Inadequate vectorization from misalignment and loop peeling
  - Large memory footprint not fitting in L1 and L2 caches
  - Memory latency bound and not saturating memory bandwidth
- Optimizations
  1. Break outer loop over j into small chunks. More thread parallelism; smaller footprint per thread.
  2. Compute using thread-private statically sized arrays. Improved vectorization.
  3. Combine/eliminate temporaries to reduce footprint from 100KB → 60KB thread. More threads/core hide memory latency.



- Host CPU and Intel Phi run identical source code
- Effort optimizing for Phi benefits host

WSM5 Microphysics

WSM5 CONUS 12KM Workload, 6.53 GF/call (Intel SDE) (www.mmm.ucar.edu/wrf/WG2/bench)



* Improved GPU/CUDA Based Parallel Weather and Research Forecast (WRF) Single Moment 5-Class (WSM5) Cloud Microphysics. J. Mielikainen, B. Huang, H-L. A. Huang, and M.D. Goldberg. IEEE JSTARS, Vol. 5, No. 4, Aug. 2012 and personal communication

Whole code performance

CONUS 12km*				***w/o wsm5 optim.***
	1 node	Xeon (2x SNB-EP, 16 cores):	109.2s	*was 123.9s*
	1 node	Intel Phi (KNC, 61 cores):	109.6s	*was 119.2s*
CONUS 2.5km*				***with and w/o Phi***
	8 nodes	2x SNB per node	328.4s	
	8 nodes	2x SNB + KNC per node	223.5s	*profit: 1.47x*

- Standard 12km and 2.5km benchmark cases
  - Times in seconds for 3 hour simulations, compute-only
  - http://www.mmm.ucar.edu/wrf/WG2/bench
- CONUS 12km (*Indraneil Gokhale, Intel Corp.)
  - Phi performance equivalent to two Xeon SNB-Eps
  - Phi-optimized WSM5 code yielded 10-13% improvement on both systems
- CONUS 2.5 (*Alexander Knyazev, Intel Corp.)
  - Comparison of run times with and without using Phi on nodes
  - Here, the Phi card adds the equivalent of another Xeon SNB-EP processor

Future work and summary

- Accelerators are on the path forward for HPC
  - Tianhe-2 (MIC) #1 and Titan (GPU) #2 Top 500
  - Are accelerator architectures stable?
  - Are the programming models effective?
  - Is performance portability possible?
- WRF results:
  - Individual kernels show promise
  - Whole code results are preliminary (MIC) or in progress (GPU)