GPU TECHNOLOGY CONFERENCE
April 4-7, 2016 | Silicon Valley

Write Once, Parallel Everywhere: OpenACC for GPUs, x86, OpenPower and Beyond

Michael Wolfe

PRESENTED BY
NVIDIA.

GPU TECHNOLOGY CONFERENCE

Performance Portability, 2012

Performance Portability with the Chapel Language

Albert Sidelnik, Saeed Maleki, María J. Garzarán, David Padua
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois, USA
{asideln2,maleki1,garzaran,padua}@illinois.edu

Bradford L. Chamberlain
Cray Inc.
Seattle, Washington, USA
bradc@cray.com

— IPDPS 2012, Shanghai, China

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6267860&tag=1

Performance Portability, 2005

Performance Portability on EARTH: A Case Study across Several Parallel Architectures

Weirong Zhu Yanwei Niu Guang R. Gao
Department of Electrical & Computer Engineering, University of Delaware
{weirong, niu, ggao}@capsl.udel.edu

Abstract

With the rapidly increasing diversity of parallel architec-

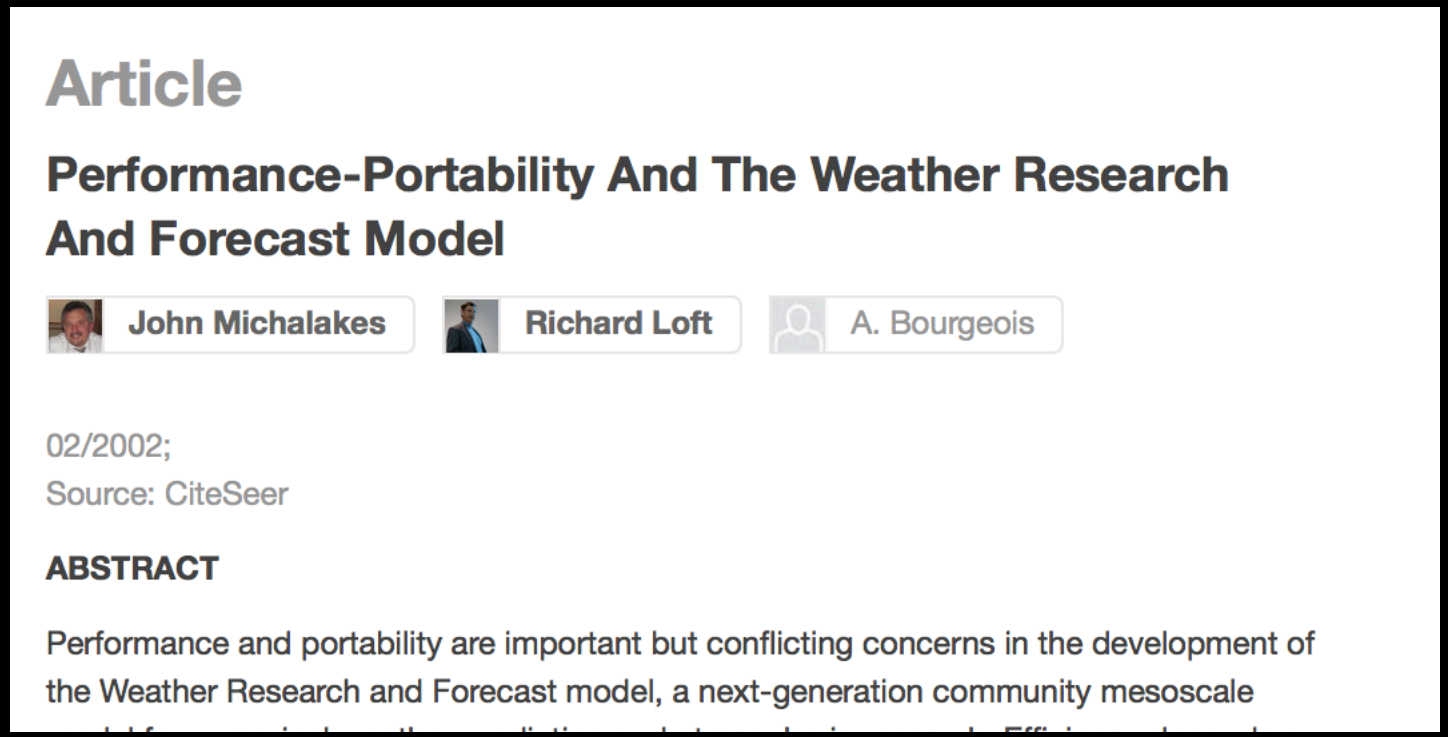
Another term for *code portability* is *compiling portability* [16]. It means that the code of parallel applications targeted to one platform can also be recompiled and executed

— IPDPS 2005, Denver, Colorado

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1420218&tag=1

GPU TECHNOLOGY CONFERENCE

Performance Portability, 2001



Article

Performance-Portability And The Weather Research And Forecast Model

John Michalakes Richard Loft A. Bourgeois

02/2002;
Source: CiteSeer

ABSTRACT

Performance and portability are important but conflicting concerns in the development of the Weather Research and Forecast model, a next-generation community mesoscale

— HPC Asia 2001,Queensland, Australia

http://www.wrf-model.org/wrfadmin/publications.php

GPU TECHNOLOGY CONFERENCE

Performance Portability, 1995

PERFORMANCE OF PARALLEL COMPUTERS FOR SPECTRAL ATMOSPHERIC MODELS

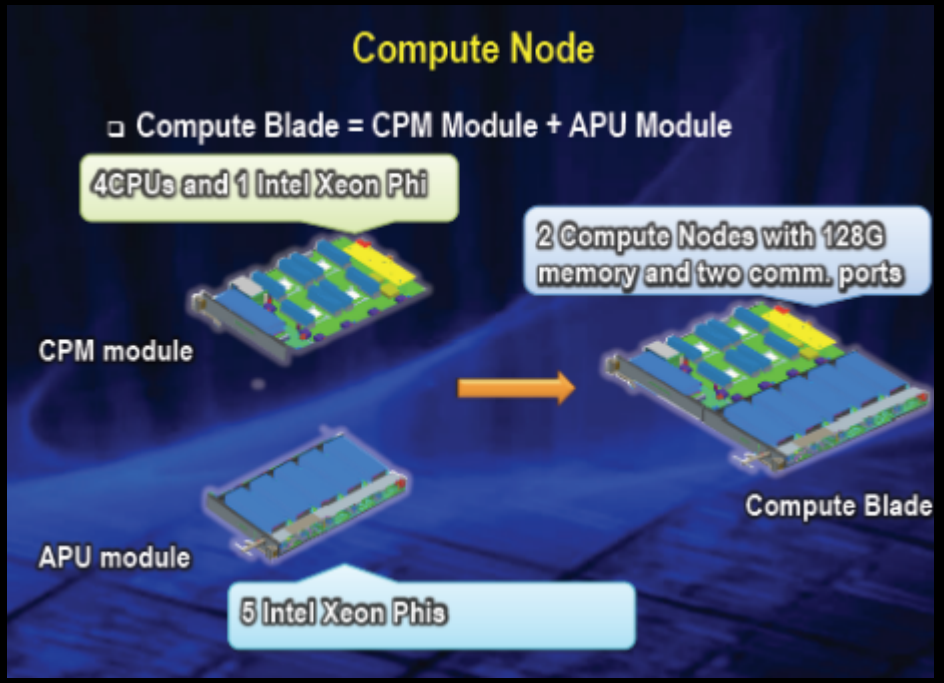
Ian T. Foster *
Brian Toonen *
Patrick H. Worley †

impact on performance. In addition, optimal choices for parallel algorithm, communication protocols, and coding style vary significantly from machine to machine. Hence, performance tuning is important both when developing a spectral model for a single computer, and when developing a model intended to operate on several different computers. We believe that the solution to this problem is to design codes that allow tuning parameters to be set at run-time. This approach supports both (performance) portability and the empirical determination of optimal parameters.

— ORNL/TM-12986, April 1995
http://www.epm.ornl.gov/~worley/papers/tmbenchmark.ps

GPU TECHNOLOGY CONFERENCE

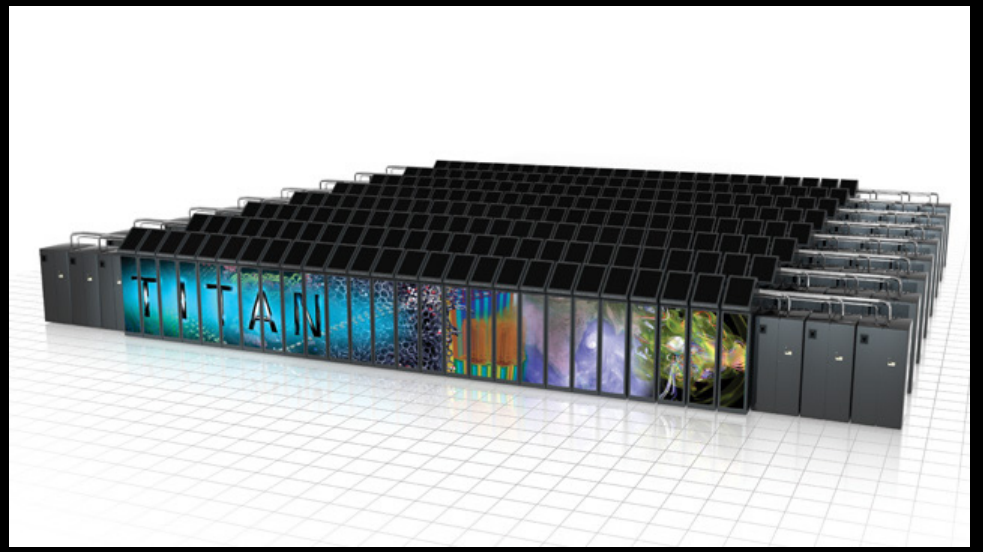
Tianhe-2 Supercomputer

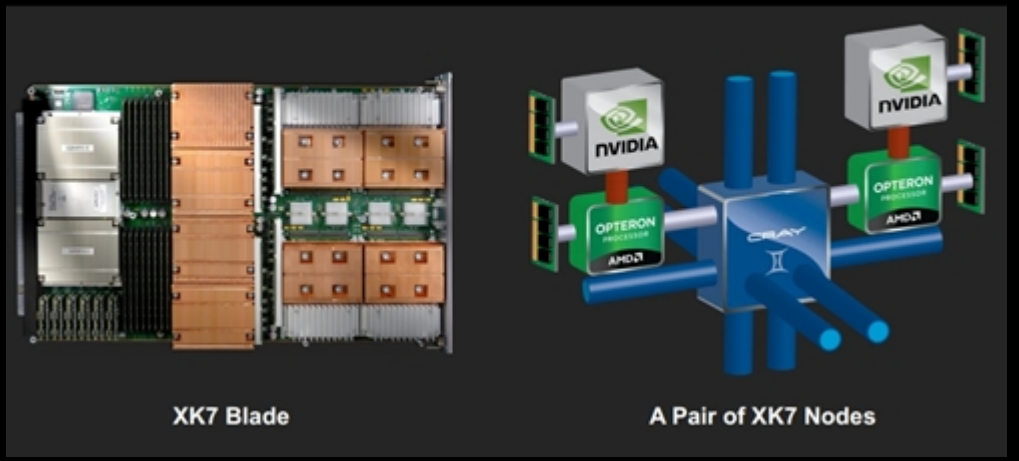


NUDT, China

GPU TECHNOLOGY CONFERENCE

Titan Supercomputer

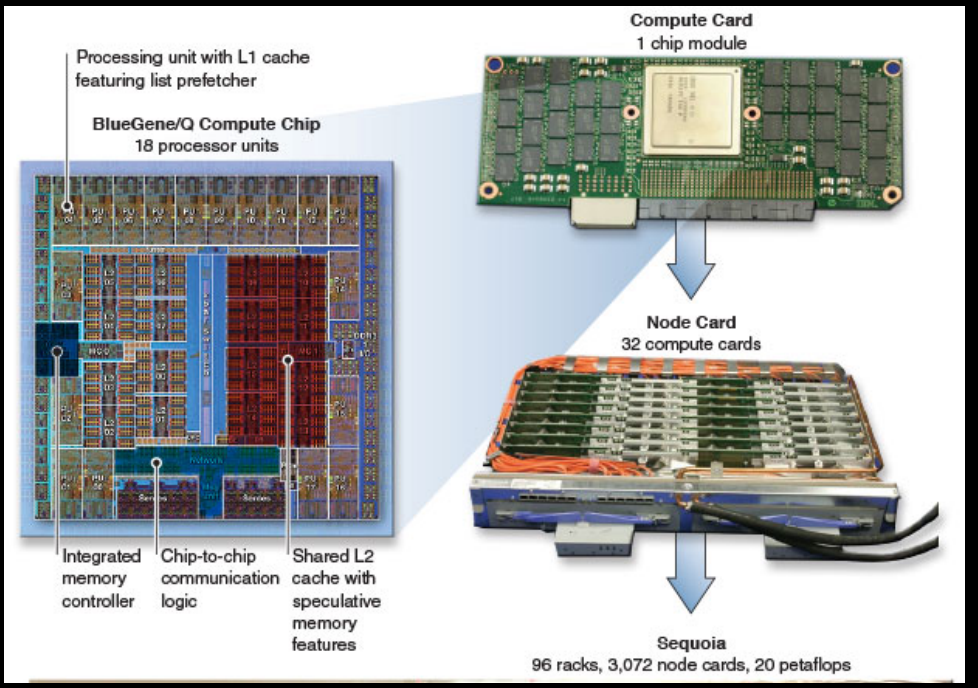




Oak Ridge National Laboratory, Tennessee

GPU TECHNOLOGY CONFERENCE

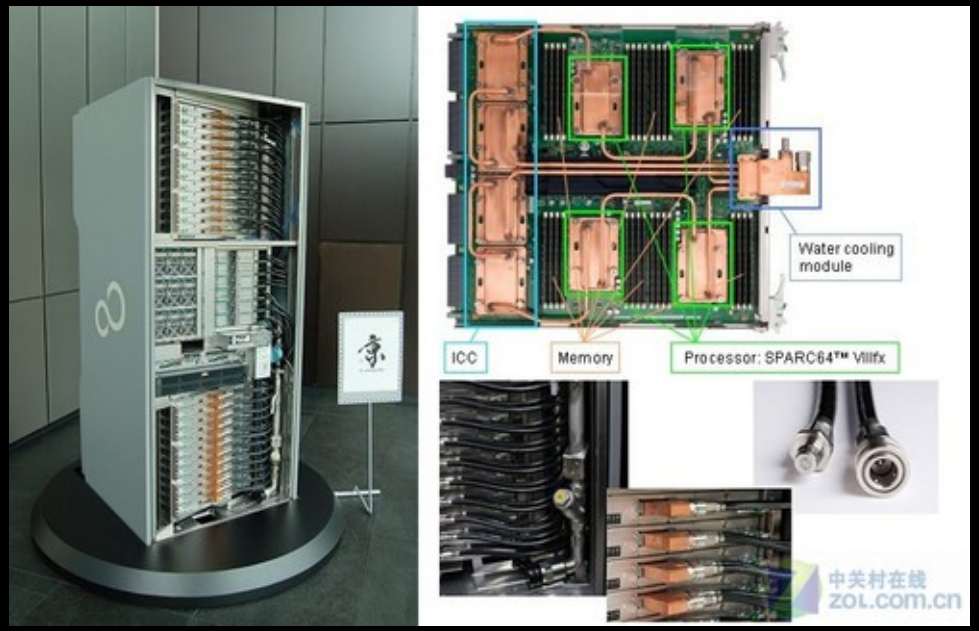
Sequoia Supercomputer



Lawrence Livermore National Laboratory, California

GPU TECHNOLOGY CONFERENCE

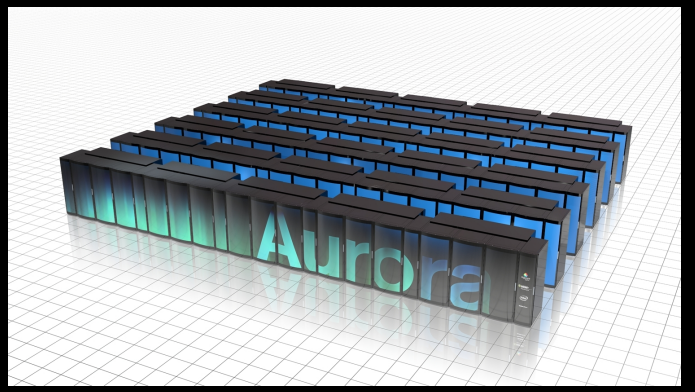
K Supercomputer



RIKEN Advanced Institute Computational Science, Kobe, Japan

GPU TECHNOLOGY CONFERENCE

CORAL Supercomputers



US Dept. of Energy, Collaboration Oak Ridge, Argonne, Livermore, 2017-8

GPU TECHNOLOGY CONFERENCE

Does OpenMP give Performance Portability?

“Even with OpenMP accelerator directives ..., two different sources are necessary”

“Most people are resigned to having different sources for different platforms”

Enabling Application Portability Across HPC Platforms: An Application Perspective

Koniges, Mattson, Foertter, He, Gerber, OpenMPCon, 2015, Aachen, Germany

http://openmpcon.org/wp-content/uploads/openmpcon2015-tim-mattson-portability.pdf

GPU TECHNOLOGY CONFERENCE

Does OpenMP give Performance Portability?

Whining about Performance Portability

"But there is a pretty darn good performance portable language. It's called OpenCL"

"Having a common code base using a portable programming environment, even if you must fill the code with if-defs or have architectural specific versions of kernels ... is the only way to support maintainability"

Tim Mattson (Intel) Presentation at OpenMPCon, 2015, Aachen, Germany

http://openmpcon.org/wp-content/uploads/openmpcon2015-tim-mattson-portability.pdf

GPU TECHNOLOGY CONFERENCE

Does OpenCL give Performance Portability?

"By creating an efficient, close-to-the-metal programming interface, OpenCL will form the foundation layer of a parallel computing ecosystem of platform-independent tools, middleware and applications."

OpenCL Specification (1.0 through 2.1) *https://www.khronos.org/opencl/*

"True, we may need to write a new version of our kernel to get the best performance on Architecture A, but isn't this what we actually want?"

OpenCL: Free Your GPU...and the rest of your system too!
HPCwire, May 6, 2013

http://www.hpcwire.com/2013/05/06/opencl_free_your_gpu_and_the_rest_of_your_system_too_/

GPU TECHNOLOGY CONFERENCE

Do Libraries give Performance Portability?

Scalapack

Thrust

Magma

BLAS

MKL

ESSL

LibSCI

...

GPU TECHNOLOGY CONFERENCE

Confidence in Performance Portability

“Where is Performance Portability”

“Titan, Mira, Edison represent 3 different architectures
- not performance portable across systems”

“Best case #1 – OpenMP4 absorbs accelerator features, but code requires ifdef”

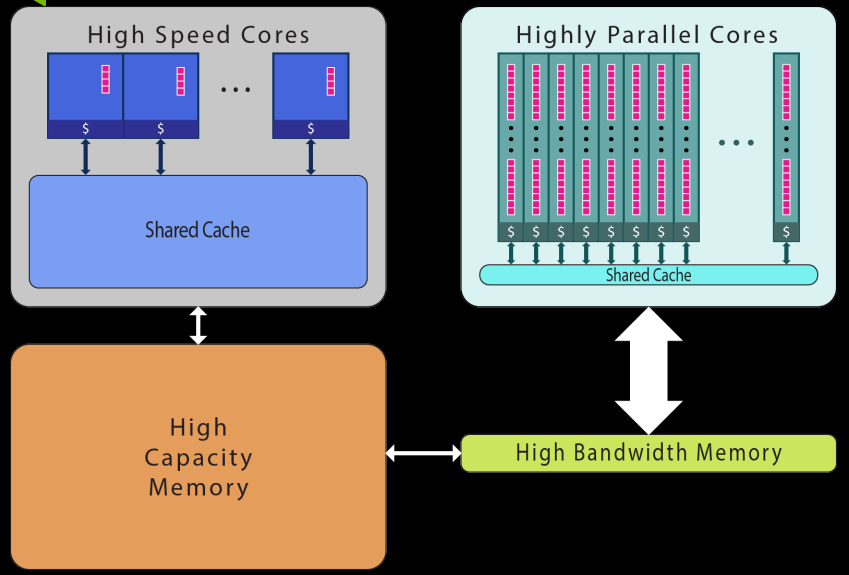
“Best case #2 – Architectures converge by 2023”

Exascale Programming Models and Environments Research
Kathy Yelick, Associate Lab Directory for Computing Sciences, LBNL
Intel Xeon Phi Users Group, October, 2015, Berkeley, California

https://drive.google.com/folderview?id=0B9kBqCR08plob0h5amlNenUydzQ&usp=sharing

GPU TECHNOLOGY CONFERENCE

Higher-Level Programming: OpenACC



© 2014 NVIDIA Corporation

```
real, allocatable :: a(:), b(:)
...
allocate(a(n),b(n))
...
!$acc data copy(a,b)
call process( a, b, n )
!$acc end data
...
subroutine process( a, b, n )
  real :: a(:), b(:)
  integer :: n, i
  !$acc parallel loop
  do i = 1, n
    b(i) = exp(sin(a(i)))
  enddo
end subroutine
```

GPU TECHNOLOGY CONFERENCE

Data directives

- Data construct
  - allocates device memory
  - moves data in/out
- Update self(b)
  - copies device->host
  - aka update host(b)
- Update device(b)
  - copies host->device

```
real, allocatable :: a(:), b(:)
...
allocate(a(n),b(n))
...
!$acc data copyin(a) copyout(b)
...
 call process( a, b, n )
...
!$acc update self(b)
 call updatehalo(b)
!$acc update device(b)
...
!$acc end data
...
```

GPU TECHNOLOGY CONFERENCE

Compute regions

- Parallel region
  - launches a device kernel
  - gangs / workers / vectors

```
subroutine process( a, b, n )
  real :: a(:,:), b(:,:)
  integer :: n, i, j
  !$acc parallel loop present(a,b)
  do j = 1, n
    !$acc loop vector
    do i = 1, n
      b(i,j) = exp(sin(a(i,j)))
    enddo
  enddo
end subroutine
```

OpenACC Performance Portability

```
!$acc kernels loop
do j = 1, m
  do i = 1, n
       a(j,i) = b(j,i)*alpha +
                c(i,j)*beta
  enddo
enddo
```

```
% pgf90 a.f90 -ta=multicore -c -Minfo
sub:
  10, Loop is parallelizable
      Generating Multicore code
      10, !$acc loop gang
  11, Loop is parallelizable
```

CPU

OpenACC

Directives for Accelerators

```
% pgf90 a.f90 -ta=tesla -c -Minfo
sub:
   9, Generating present(a(:,:),b(:,:),c(:,:))
  10, Loop is parallelizable
  11, Loop is parallelizable
      Accelerator kernel generated
      Generating Tesla code
      10, !$acc loop gang, vector(4)
      11, !$acc loop gang, vector(32)
```

GPU

GPU TECHNOLOGY CONFERENCE

Using the PGI compilers

pgfortran, pgc++, pgcc

-acc

default -ta=tesla,host

-ta=tesla[:suboptions...]

-ta=radeon[:suboptions...]

-ta=multicore

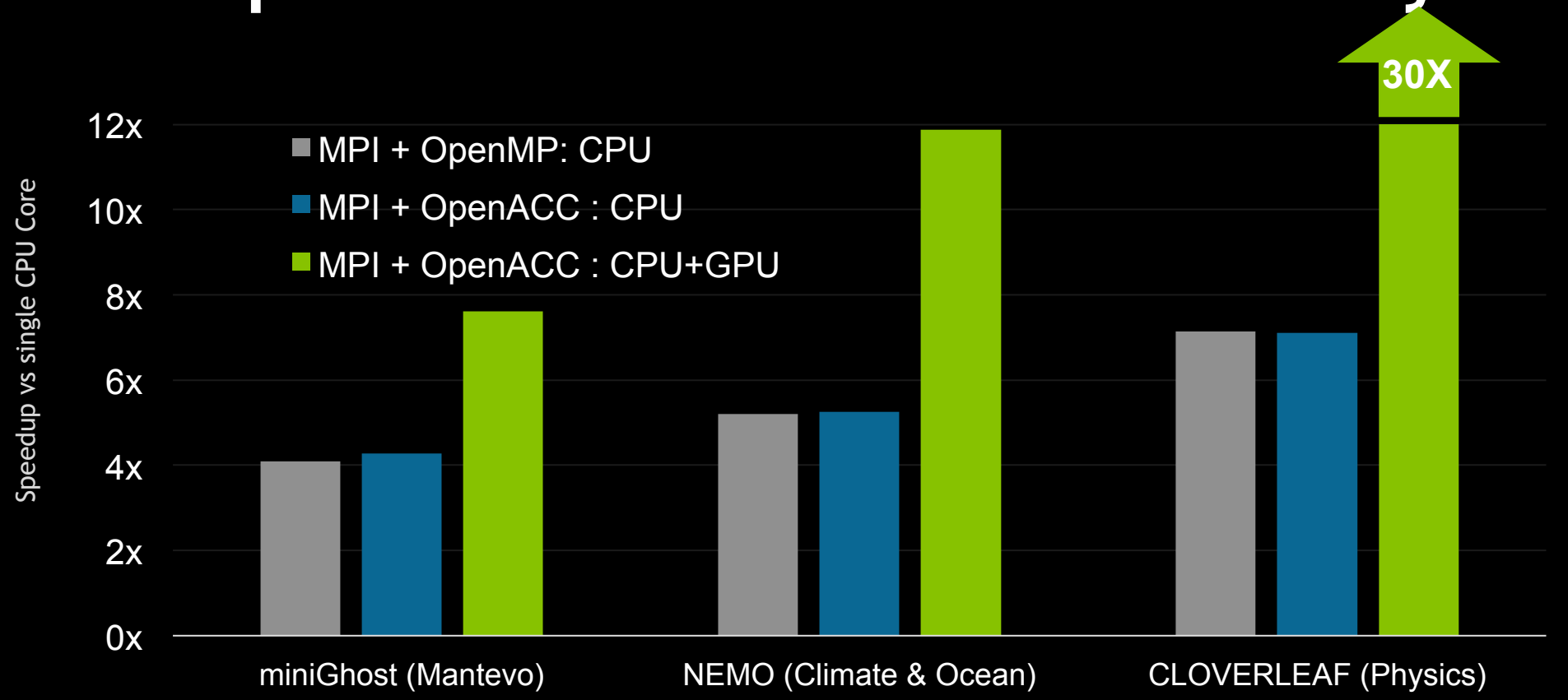
-Minfo=accel

```
% pgfortran -ta=tesla a.f90 -Minfo=accel
% ./a.out

% pgfortran -acc -c b.f90 -Minfo=accel
% pgfortran -acc -c c.f90 -Minfo=accel
% pgfortran -acc -o c.exe b.o c.o
% ./c.exe
```

GPU TECHNOLOGY CONFERENCE

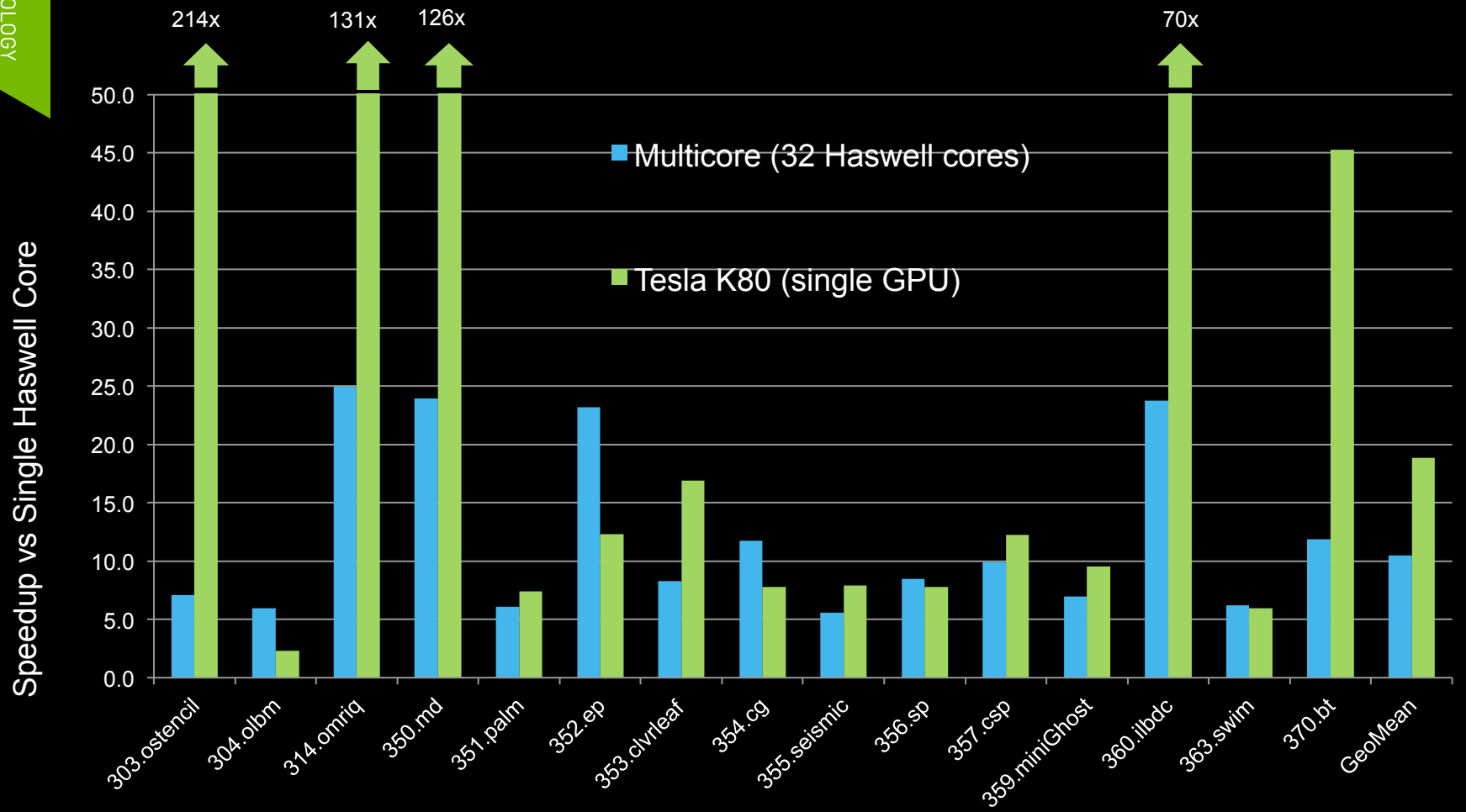
OpenACC Performance Portability



359.miniGhost: CPU: Intel Xeon E5-2698 v3, 2 sockets, 32-cores total, GPU: Tesla K80 (single GPU)
NEMO: Each socket CPU: Intel Xeon E5---2698 v3, 16 cores; GPU: NVIDIA K80 both GPUs
CLOVERLEAF: CPU: Dual socket Intel Xeon CPU E5-2690 v2, 20 cores total, GPU: Tesla K80 both GPUs

GPU TECHNOLOGY CONFERENCE

SPECAccel Performance Portability



System Information:
Supermicro SYS-2028GR-TRT
CPU: Intel Xeon E5-2698 v3, 2 sockets, 32 cores, HT disabled
GPU: NVIDIA Tesla K80 (single GPU)
OS: CentOS 6.6
Compiler: PGI 16.1

PGI 16.1 OpenACC Multicore and K80 results from SPEC ACCEL™ measured Mar 2016. SPEC® and the benchmark name SPEC ACCEL™ are registered trademarks of the Standard Performance Evaluation Corporation.

GPU TECHNOLOGY CONFERENCE

DEMAND PERFORMANCE PORTABILITY!

Don’t settle for less! You deserve better!