Running OpenACC Programs on NVIDIA and AMD GPUs

Michael Wolfe, Compiler Engineer
The Portland Group
www.pgroup.com
What is OpenACC?

A set of directive-based extensions to C, C++ and Fortran that allow you to annotate regions of code and data for offloading from a CPU host to an attached Accelerator

NVIDIA Kepler Overall Block Diagram*

NVIDIA Kepler SMX Block Diagram*

- 192 SP CUDA cores
- 64 DP units
- 32 SFUs
- 32 ld/st units

AMD Radeon 7970 Compute Unit*

CPU+Accelerator
Abstract Machine Architecture
OpenACC Directives

```c
#pragma acc data copyin(in[0:n]) copyout(out[0:n]) \ 
    copy(force[0:n], vel[0:n])
{
    #pragma acc parallel loop
    for (int i = 0; i < n; i++)
    {
        . . . // update forces
    }
    #pragma acc parallel loop
    for (int i = 0; i < n; i++)
    {
        . . . // update positions, velocities
    }
}
```
Building OpenACC Programs

% pgcc -acc -ta=nvidia -c foo.c
% pgcc -acc -ta=nvidia -o foo.exe foo.o
% foo.exe

% pgcc -acc -ta=radeon -c bar.c
% pgcc -acc -ta=radeon -o bar.exe bar.o
% bar.exe

% pgcc -help -ta
OpenACC Features

- Single source code for CPU and GPU
- Offload data and loops with directives
- Incrementally tune data movement
- Overlap data movement with computation
- Re-use Accelerator data across kernels, even across procedure calls
- Easy to experiment with alternative loop schedules, mapping of parallelism to HW
OpenACC 2.0
Upcoming Features

- Procedure calls on the Accelerator
- Unstructured Accelerator data lifetimes
- Nested parallelism
- Atomic operations
- Better interaction with OpenMP parallelism
- and more...
Running OpenACC Programs on NVIDIA and AMD GPUs

-acc -ta=nvidia -ta=radeon

Accelerators exploit parallelism, regularity

- expose, express, exploit
  - algorithm
  - language
  - compiler + runtime + hardware

www.pgroup.com/openacc
www.openacc.org