OPENACC AND UNIFIED MEMORY
Doug Miles, Cray User Group Meeting, 10 May 2017
OPENACC DIRECTIVES
Data directives are designed to be optional

Manage Data Movement
Initiate Parallel Execution
Optimize Loop Mappings

#pragma acc data copyin(a,b) copyout(c)
{
    ...
    #pragma acc parallel
    {
        #pragma acc loop gang vector
        for (i = 0; i < n; ++i) {
            c[i] = a[i] + b[i];
            ...
        }
    }
    ...
}
Data directives are designed to be optional

```c
#pragma acc data copyin(a,b) copyout(c)
{
    ...
    #pragma acc parallel
    {
        #pragma acc loop gang vector
        for (i = 0; i < n; ++i) {
            c[i] = a[i] + b[i];
            ...
        }
    }
    ...
}
```
OPENACC DIRECTIVES
Data directives are designed to be optional

... #pragma acc parallel {
  #pragma acc loop gang vector
  for (i = 0; i < n; ++i) {
    c[i] = a[i] + b[i];
    ...
  }
} ...

PGI
OPENACC FOR MULTICORE CPUs & GPUs

% pgfortran -ta=multicore -fast -Minfo=acc -c \ update_tile_halo_kernel.f90

... 100, Loop is parallelizable
   Generating Multicore code
100, !$acc loop gang
102, Loop is parallelizable

% pgfortran -ta=tesla -fast -Minfo=acc -c \ update_tile_halo_kernel.f90

... 100, Loop is parallelizable
102, Loop is parallelizable
   Accelerator kernel generated
   Generating Tesla code
100, !$acc loop gang, vector(4) ! blockIdx%x threadIdx%x
102, !$acc loop gang, vector(32) ! blockIdx%x threadIdx%x
OpenACC SPEC ACCEL 1.1 Benchmarks
Geometric mean across all 15 benchmarks

Performance measured February and March, 2017 and are considered estimates per SPEC run and reporting rules. SPEC® and SPEC ACCEL® are registered trademarks of the Standard Performance Evaluation Corporation (www.spec.org).
CUDA UNIFIED MEMORY FOR TESLA

Servicing CPU and GPU Page Faults

GPU Code

```c
__global__
Void setValue(char *ptr, int index, char val)
{
    ptr[index] = val;
}
```

CPU Code

```c
cudaMallocManaged(&array, size);
memset(array, size);
setValue<<<...>>>(array, size/2, 5);
```

GPU Memory Mapping

![GPU Memory Mapping Diagram]

CPU Memory Mapping

![CPU Memory Mapping Diagram]
OPENACC WITH CUDA UNIFIED MEMORY
P100 Paging Engine Moves All Dynamically Allocated Data

100% = Pure Directive-based Data Movement

PGI 17.1 Compilers OpenACC SPEC ACCEL™ 1.1 performance measured March, 2017 SPEC® and the benchmark name SPEC ACCEL™ are registered trademarks of the Standard Performance Evaluation Corporation.
TESLA GPU PROGRAMMING IN 3 STEPS

PARALLELIZE
Parallelize with OpenACC for multicore CPUs

% pgc++ -ta=multicore ...

while ( error > tol && ...
   error = 0.0;
   #pragma acc parallel loop ...
   for( int j = 1; ... 
   #pragma acc loop
      for( int i = 1; ... 
      ...
   }

OFFLOAD
Port to Tesla using OpenACC with CUDA Unified Memory

OPTIMIZE
Optimize and overlap data movement using OpenACC data directives
**TESLA GPU PROGRAMMING IN 3 STEPS**

### PARALLELIZE

Parallelize with OpenACC for multicore CPUs

```bash
% pgc++ -ta=multicore ...
```

```c
while ( error > tol && ...) {
    error = 0.0;
    #pragma acc parallel loop ...
    for( int j = 1; ...
        #pragma acc loop
        for( int i = 1; ...
            ...
    }
}
```

### OFFLOAD

Port to Tesla using OpenACC with CUDA Unified Memory

```bash
% pgc++ -ta=tesla:managed ...
```

```c
while ( error > tol && ...) {
    error = 0.0;
    #pragma acc parallel loop ...
    for( int j = 1; ...
        #pragma acc loop
        for( int i = 1; ...
            ...
    }
}
```

### OPTIMIZE

Optimize and overlap data movement using OpenACC data directives

```bash
% pgc++ -ta=tesla:managed ...
```
TESLA GPU PROGRAMMING IN 3 STEPS

PARALLELIZE

Parallelize with OpenACC for multicore CPUs

% pgc++ -ta=multicore ...

while ( error > tol && …
  error = 0.0;
#pragma acc parallel loop …
  for( int j = 1; …
#pragma acc loop
      for( int i = 1; …
        ...
  )
...

OFFLOAD

Port to Tesla using OpenACC with CUDA Unified Memory

% pgc++ -ta=tesla:managed ...

while ( error > tol && …
  error = 0.0;
#pragma acc parallel loop …
  for( int j = 1; …
#pragma acc loop
      for( int i = 1; …
        ...
  )
...

OPTIMIZE

Optimize and overlap data movement using OpenACC data directives

#pragma acc data create ...
while ( error > tol && …
  error = 0.0;
#pragma acc parallel loop ...
  for( int j = 1; …
#pragma acc loop
      for( int i = 1; …
        ...
  )
...
PGI® COMPILERS & TOOLS