

May 3, 2013

# BLUE WATERS

SUSTAINED PETASCALE COMPUTING

OpenACC compiling and  
performance tips



GREAT LAKES CONSORTIUM  
FOR PETASCALE COMPUTATION

CRAY®

## OpenACC compiler support

Cray

Module load PrgEnv-cray craype-accel-nvidia35

- Fortran
  - -h acc, noomp # openmp is enabled by default, be careful mixing
  - -fpic -dynamic
  - -rm # include a .lst listing file to show the loop markup
  - -G2 # -g has been observed to break Cray OpenACC code
- C
  - -h pragma=acc -h nopragma=omp
  - -fpic -dynamic
  - -h msgs # show loop markup in stdout/stderr
  - -Gp # bonus points to the person who synchronizes Cray compiler flags between fortran and c...

## Cray -rm # loop mark

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> ftn -h acc -rm -c push2.f
```

```
!$acc parallel num_gangs(1) vector_length(3072)
ftn-7271 crayftn: WARNING GPUSH2L, File = push2.f, Line = 145
Unsupported OpenACC vector_length expression: Converting 3072 to 1024.
```

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> grep --after-context=5 '!$acc parallel num_gangs(1)
vector_length(3072)' push2.lst
145. + G-----< !$acc parallel num_gangs(1) vector_length(3072)
ftn-7271 ftn: WARNING File = push2.f, Line = 145
Unsupported OpenACC vector_length expression: Converting 3072 to 1024.
```

```
146.      G          !!$acc kernels
147.      G          !!data copy(part),copyin(fxy),create(nn,mm,dxp,dyp,np,mp,dx,dy,vx,vy)
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> grep 'line 145 ' push2.lst
A region starting at line 145 and ending at line 240 was placed on the accelerator.
```

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f>
```

## OpenACC compiler support

### PGI

Module load PrgEnv-pgi cudatoolkit

- Cudatoolkit is required, PGI is creating CUDA code as intermediate
  - -ta=nvidia,keepgpu,keepptx
- Fortran , C # nice
  - -acc -ta=nvidia
  - -mcmodel=medium
  - -Minfo=accel

### GNU

- Don't touch that dial!

## PGI -Minfo=accel

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> ftn -acc -ta=nvidia -Minfo=accel -c push2.f
gpush21:
 145, Accelerator kernel generated
 145, CC 1.3 : 18 registers; 112 shared, 32 constant, 0 local memory bytes
      CC 2.0 : 26 registers; 0 shared, 132 constant, 0 local memory bytes
 148, !$acc loop vector(3072) ! blockidx%x threadidx%x
 169, Sum reduction generated for sum1
145, Generating present_or_copy(part(:, :, :, :))
    Generating present_or_copyin(fxy(:, :, :, :))
    Generating compute capability 1.3 binary
    Generating compute capability 2.0 binary
148, Loop is parallelizable
```



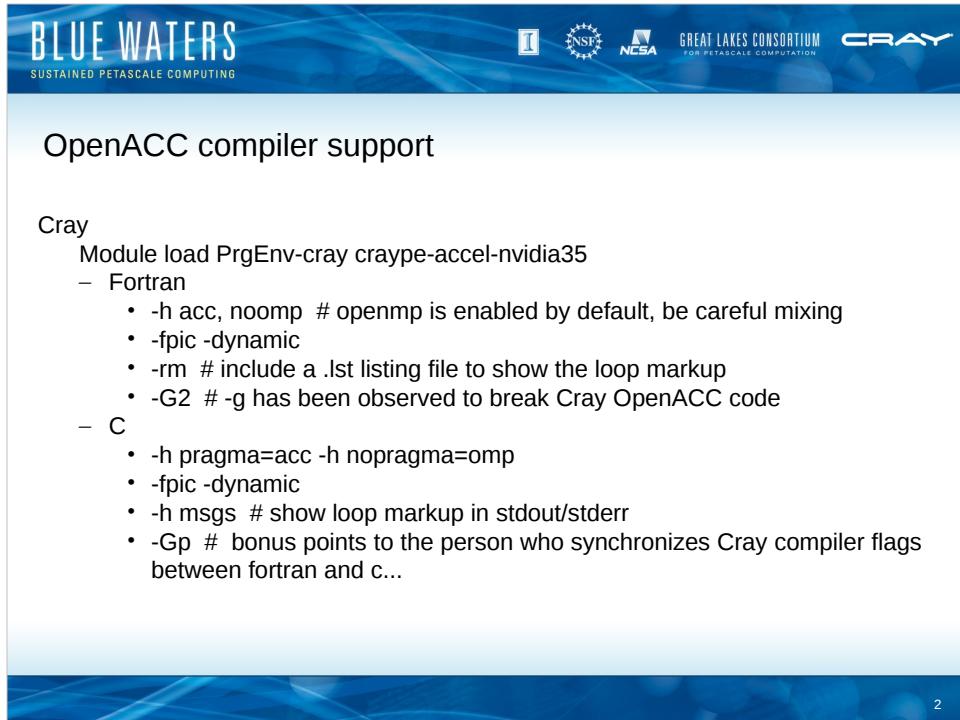
The slide features a blue gradient background with abstract wavy patterns. In the top left corner, the text "BLUE WATERS" is displayed in large, white, sans-serif capital letters. Below it, the words "SUSTAINED PETASCALE COMPUTING" are written in a smaller, yellow, all-caps font. In the top right corner, the date "May 3, 2013" is shown in a small, white, sans-serif font. In the center-left area, the text "OpenACC compiling and performance tips" is presented in a black, sans-serif font. At the bottom of the slide, there is a dark blue horizontal bar containing several logos: a white square with a vertical line, the NCSA logo (a white square with a diagonal line), the NSF logo (a circular emblem with stars and the acronym "NSF"), the text "GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION" in white, and the CRAY logo (the word "CRAY" in a white, bold, sans-serif font).

OpenACC compiling and performance tips

May 3, 2013

OpenACC compiling and performance tips

I NCSA NSF GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION CRAY



The slide features the "BLUE WATERS SUSTAINED PETASCALE COMPUTING" logo at the top left. To the right are logos for I, NSF, NCSA, GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION, and CRAY. The main title "OpenACC compiler support" is centered above a detailed list of compiler flags for Cray's PrgEnv environment.

Cray

Module load PrgEnv-cray craype-accel-nvidia35

- Fortran
  - -h acc, noomp # openmp is enabled by default, be careful mixing
  - -fpic -dynamic
  - -rm # include a .lst listing file to show the loop markup
  - -G2 # -g has been observed to break Cray OpenACC code
- C
  - -h pragma=acc -h nopragma=omp
  - -fpic -dynamic
  - -h msgs # show loop markup in stdout/stderr
  - -Gp # bonus points to the person who synchronizes Cray compiler flags between fortran and c...

2

Watch out for the differences in flags when using the Cray compiler environment.

Omitting craype-accel-nvidia35 can result in silent failure mode:

```
ftn -h acc -c myfile.c
```

...

```
ldd a.out
```

not a dynamic executable

...you've made an executable that will not use the accelerator at runtime, but the code will probably run .



## Cray -rm # loop mark

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> ftn -h acc -rm -c push2.f

!$acc parallel num_gangs(1) vector_length(3072)
ftn-7271 crayftn: WARNING GFUSH2L, File = push2.f, Line = 145
Unsupported OpenACC vector_length expression: Converting 3072 to 1024.

arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> grep --after-context=5 '!$acc parallel num_gangs(1)
vector_length(3072)' push2.lst
145. + G-----< !$acc parallel num_gangs(1) vector_length(3072)
ftn-7271 ftn: WARNING File = push2.f, Line = 145
Unsupported OpenACC vector_length expression: Converting 3072 to 1024.

146.    G          !$acc kernels
147.    G          !!data copy(part),copyin(fxy),create(nn,mm,dxp,dyp,np,mp,dx,dy,vx,vy)
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> grep 'line 145' push2.lst
A region starting at line 145 and ending at line 240 was placed on the accelerator.

arnoldg@h2ologin2:~/Mori/pic2.0-acc-f>
```

Cray loop mark .lst listings are a good source of information about how the compiler optimized (or could not optimize) your code. Along with basic profiling info, this is a good starting point for code optimization.

## OpenACC compiler support

PGI

- Module load PrgEnv-pgi cudatoolkit
  - Cudatoolkit is required, PGI is creating CUDA code as intermediate
    - ta=nvidia,keepgpu,keepptx
  - Fortran , C # nice
    - acc -ta=nvidia
    - mcmodel=medium
    - Minfo=accel

GNU

- Don't touch that dial!

4

Omitting the cudatoolkit module will result in silent failure mode (program compiles and links with possibly just a warning):

```
/opt/pgi/12.10.0/linux86-64/12.10/lib/libacc1mp.a(nvinitx.o): In function  
`__pgi_cu_init_y':  
/usr/pgrel/extract/x86/2012/rte/accel/hammer/lib-linux86-64mp/..src-  
nv/nvinitx.c:238: warning: Using 'dlopen' in statically linked  
applications requires at runtime the shared libraries from the glibc  
version used for linking
```

Keepgpu and keepptx will preserve the intermediate cuda and/or Nvidia ptx assembly files. This is in contrast to the Cray compiler that compiles directly to ptx assembly and creates no intermediate cuda source.

mcmodel=medium allows for larger static memory ( > 2 GB )

GNU may pickup OpenACC support once it becomes part of a future OpenMP standard.



## PGI -Minfo=accel

```
arnoldg@h2ologin2:~/Mori/pic2.0-acc-f> ftn -acc -ta=nvidia -Minfo=accel -c push2.f
gpush2l:
145, Accelerator kernel generated
    145, CC 1.3 : 18 registers; 112 shared, 32 constant, 0 local memory bytes
        CC 2.0 : 26 registers; 0 shared, 132 constant, 0 local memory bytes
    148, !$acc loop vector(3072) ! blockidx%x threadidx%x
    169, Sum reduction generated for suml
145, Generating present_or_copy(part(:,nop))
    Generating present_or_copyin(fxy(:,:,))
    Generating compute capability 1.3 binary
    Generating compute capability 2.0 binary
148, Loop is parallelizable
```

Similar to the Cray loopmark report, PGI compilers can emit detailed information about optimizations. -Minfo=accel will add comments for each OpenACC directive found and what was done (or not done) with that code region.