NVIDIA GTC

CONNECT WITH THE EXPERTS

Directive-Based GPU Programming with OpenACC

(CWE21815)

Stefan Maintz, Senior Development Technology Engineer, NVIDIA
Jeff Larkin, Senior DevTech Software Engineer, NVIDIA
Alexey Romanenko, Senior Developer Technology Engineer, NVIDIA
Markus Wetzstein, HPC Development Technology Engineer, NVIDIA
Vishal Mehta, Developer Technology, NVIDIA
Louis Stuber, Compute Developer Technology Engineer, NVIDIA
Andreas Hehn, Developer Technology Engineer, NVIDIA
Julia Levites, Senior Product Manager

NVIDIA GTC

OpenACC Directives

a directive-based parallel programming model designed for usability, performance, and portability

3 OF TOP 5 HPC

Intersect360 RESEARCH

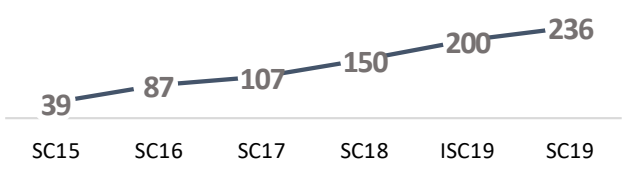
18% OF INCITE AT SUMMIT

summit

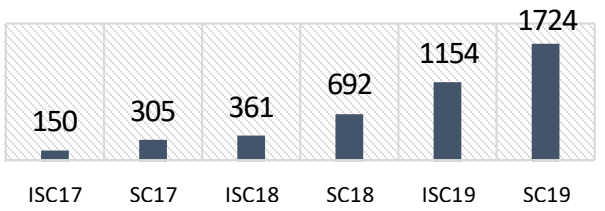
PLATFORMS SUPPORTED

NVIDIA GPU
X86 CPU
POWER CPU
Sunway
ARM CPU
AMD GPU

OPENACC APPS



OPENACC SLACK MEMBERS



>200K DOWNLOADS

PGI® Community EDITION

OPENACC Resources

Guides • Talks • Tutorials • Videos • Books • Spec • Code Samples • Teaching Materials • Events • Success Stories • Courses • Slack • Stack Overflow

FREE Compilers

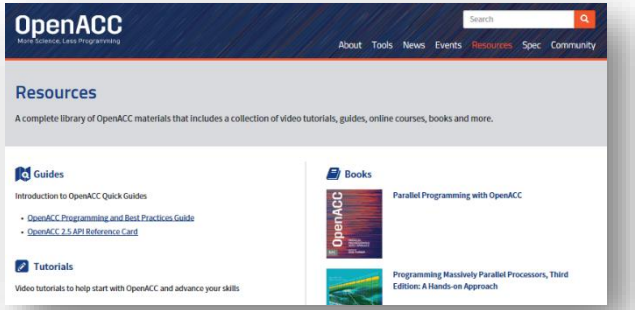
PGI Community Edition

slack

https://www.openacc.org/community#slack

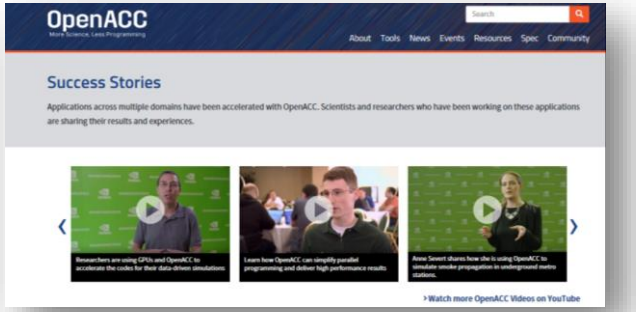
Resources

https://www.openacc.org/resources



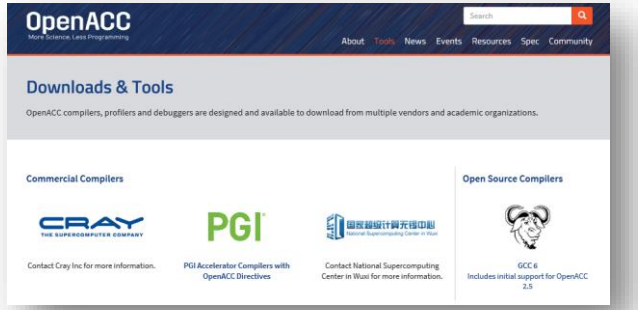
Success Stories

https://www.openacc.org/success-stories



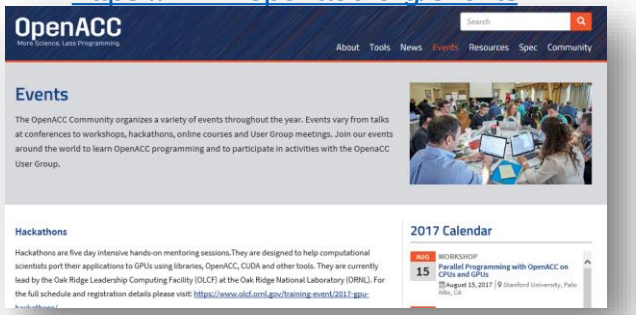
Compilers and Tools

https://www.openacc.org/tools



Events

https://www.openacc.org/events



NVIDIA GTC

Frequently Asked Questions

(CWE21815)

1. What do I need to accelerate my code with OpenACC? Is it free?

OpenACC is free to use, all you need is a C, C++ or Fortran application and an OpenACC compiler, like PGI or GCC.

2. What are the benefits of using OpenACC?

OpenACC allows you to use a single source code to run on your CPU (serial and parallel), GPU, or other parallel processor.

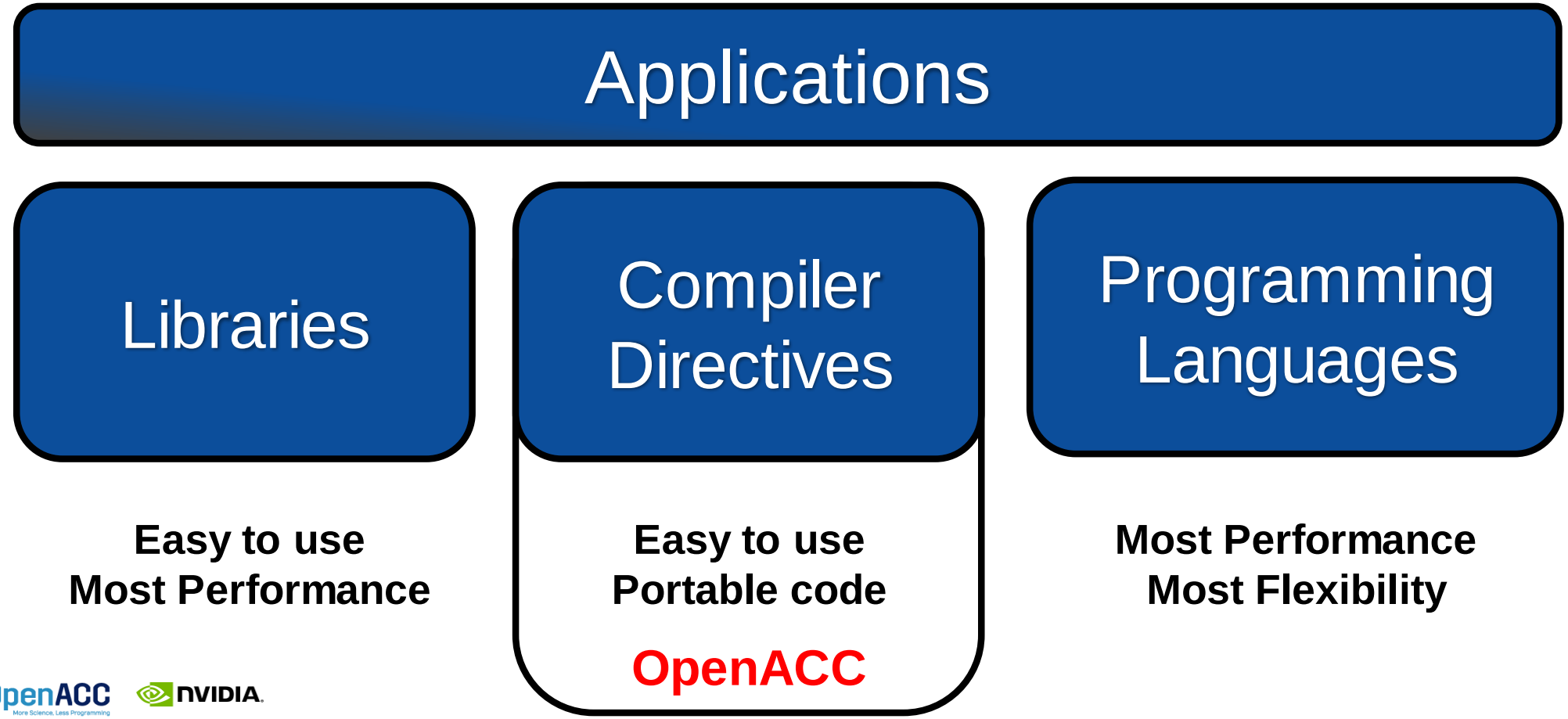
3. How interoperable is OpenACC with other frameworks or libraries?

OpenACC works well with CUDA, GPU libraries, MPI, and other frameworks.

Back-up Material

INTRODUCTION TO OPENACC

3 WAYS TO ACCELERATE APPLICATIONS



OpenACC More Science, Less Programming NVIDIA

OPENACC IS…

a directives-based **parallel programming model** designed for **performance** and **portability**.

Add Simple Compiler Directive

```
main()
 {
  <serial code>
  #pragma acc kernels
  {
    <parallel code>
  }
}
```

OpenACC Directives

Manage Data Movement

Initiate Parallel Execution

Optimize Loop Mappings

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

OpenACC
Directives for Accelerators

- Incremental
- Single source
- Interoperable
- Performance portable
- CPU, GPU, Manycore

OPENACC STRENGTHS

Incremental

- Maintain existing sequential code
- Add annotations to expose parallelism
- After verifying correctness, annotate more of the code

Single Source

- Rebuild the same code on multiple architectures
- Compiler determines how to parallelize for the desired machine
- Sequential code is maintained

Low Learning Curve

- OpenACC is meant to be easy to use, and easy to learn
- Programmer remains in familiar C, C++, or Fortran
- No requirement to learn low-level details of the hardware.

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

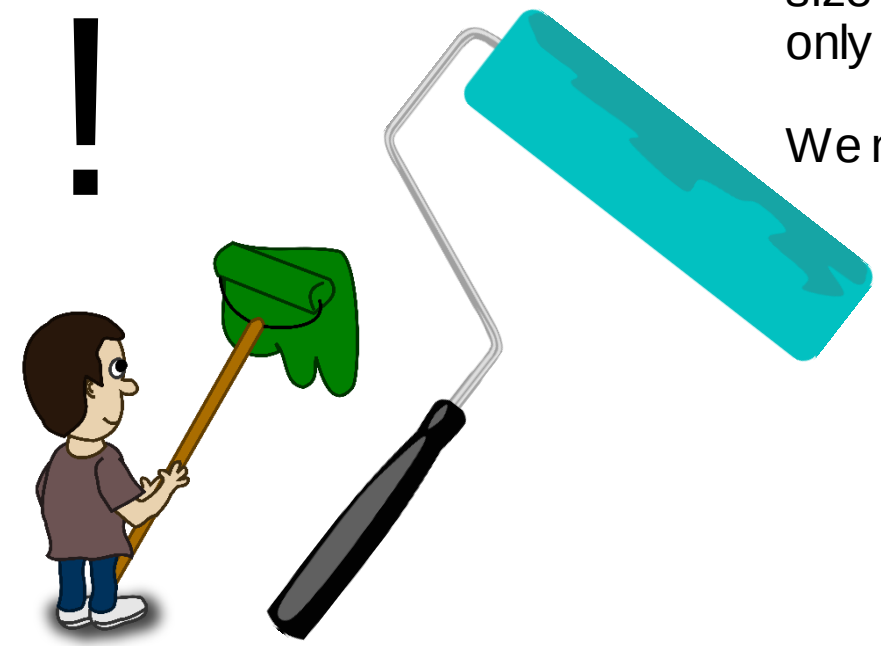
GANGS, WORKERS, AND VECTORS DEMYSTIFIED

!

How much work 1 worker can do is limited by his speed.

A single worker can only move so fast.

GANGS, WORKERS, AND VECTORS DEMYSTIFIED



Even if we increase the size of his roller, he can only paint so fast.

We need more workers!

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

Multiple workers can do more work and share resources, if organized properly.

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

By organizing our workers into groups (gangs), they can effectively work together within a floor.

Groups (gangs) on different floors can operate independently.

Since gangs operate independently, we can use as many or few as we need.

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

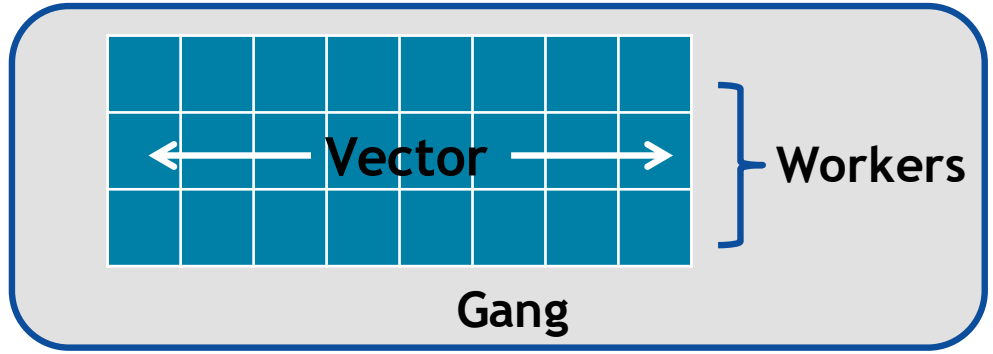
Even if there's not enough gangs for each floor, they can move to another floor when ready.

GANGS, WORKERS, AND VECTORS DEMYSTIFIED

Our painter is like an OpenACC ***worker***, he can only do so much.

His roller is like a ***vector***, he can move faster by covering more wall at once.

Eventually we need more workers, which can be organized into ***gangs*** to get more done.



LOOP OPTIMIZATION RULES OF THUMB

- It is rarely a good idea to set the number of gangs in your code, let the compiler decide.
- Most of the time you can effectively tune a loop nest by adjusting only the vector length.
- It is rare to use a worker loop on NVIDIA GPUs. When the vector length is very short, a worker loop can increase the parallelism in your gang (thread block).
- When possible, the vector loop should step through your arrays consecutively (stride==1)
- Gangs should come from outer loops, vectors from inner

INTEROPERABILITY EXAMPLES

- If you would like some more full code examples of OpenACC interoperability, follow the github link below to view a repository that contains many of the codes discussed in this module.
- If you would like to read some additional information about the concepts covered today, follow the second link to an NVIDIA devblog about OpenACC interoperability.

https://github.com/jefflarkin/openacc-interoperability

https://devblogs.nvidia.com/3-versatile-openacc-interoperability-techniques/

OpenACC
More Science, Less Programming