OPENACC TECHNICAL COMMITTEE UPDATE

Jeff Larkin <jlarkin@nvidia.com>,

Sr. DevTech SW Engineer, NVIDIA; OpenACC Technical Committee Chair

OpenACC
More Science, Less Programming

OpenACC 3.0
Released November 2019

Added C18, C++17, Fortran 2018 as supported base languages

Support for C++ lambdas

Improved multi-device support through direct memory copies and synchronization

Added zero-on-create to data clauses

Expanded list of directives that support the “if” clause

Lots of clarifications and clean-up

OpenACC

Post 3.0 Activities

OpenACC 3.0 was potentially a big change for implementers, it will take time to become widely-supported

General Clean-Up

- Restructured compute constructs & restrictions for less repetition
- Clarified ambiguities that could lead to divergent implementations
- General spec readability & clarity enhancements

Language Modernization

We support F18, C++17, and C18, but with restrictions. What can we unrestrict?

Fortran:

- Block construct
- Do concurrent

C/C++:

- Range-based for loops

Ongoing Work

*Not everything is listed here, **Nothing here is guaranteed

Improved Error Handling

Extended Memory allocation (pagelocked, unified, etc.)

Async on multicore (tasking?)

Extended Fortran bindings

Aliasing of data clauses

OPENACC FUTURE DIRECTIONS

Jeff Larkin <jlarkin@nvidia.com>,

Sr. DevTech SW Engineer, NVIDIA; OpenACC Technical Committee Chair

OpenACC
More Science, Less Programming

Ongoing Work

*Not everything is listed here, **Nothing here is guaranteed

Improved Error Handling

Extended Memory allocation (pagelocked, unified, etc.)

Async on multicore (tasking?)

Extended Fortran bindings

Aliasing of data clauses

What next for OpenACC

C++17, 20, and beyond

- How do we interface with language-level parallelism?
- What gaps should we be filling?
- Where is OpenACC no longer needed?

Fortran2018, 202X, 202Y

- Does DO CONCURRENT sufficiently meet programmer needs?
- Does data management belong in the language or remain in directives?
- What about co-arrays?

What Do You Still Need?

What challenges are you still facing that we can fix?

Are directives still relevant as languages become parallel?

What are your hopes for OpenACC?