OAK RIDGE National Laboratory

Clacc 2021: An Update on OpenACC Support for Clang and LLVM

Joel E. Denny, Seyong Lee, Jeffrey S. Vetter

Oak Ridge National Laboratory

https://csmd.ornl.gov/project/clacc

November 17, 2021: LLVM Developers' Meeting

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Acknowledgement

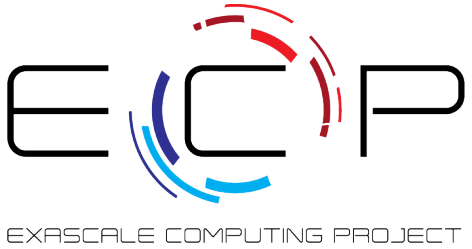
This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

This research used resources of the Experimental Computing Laboratory (ExCL) at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

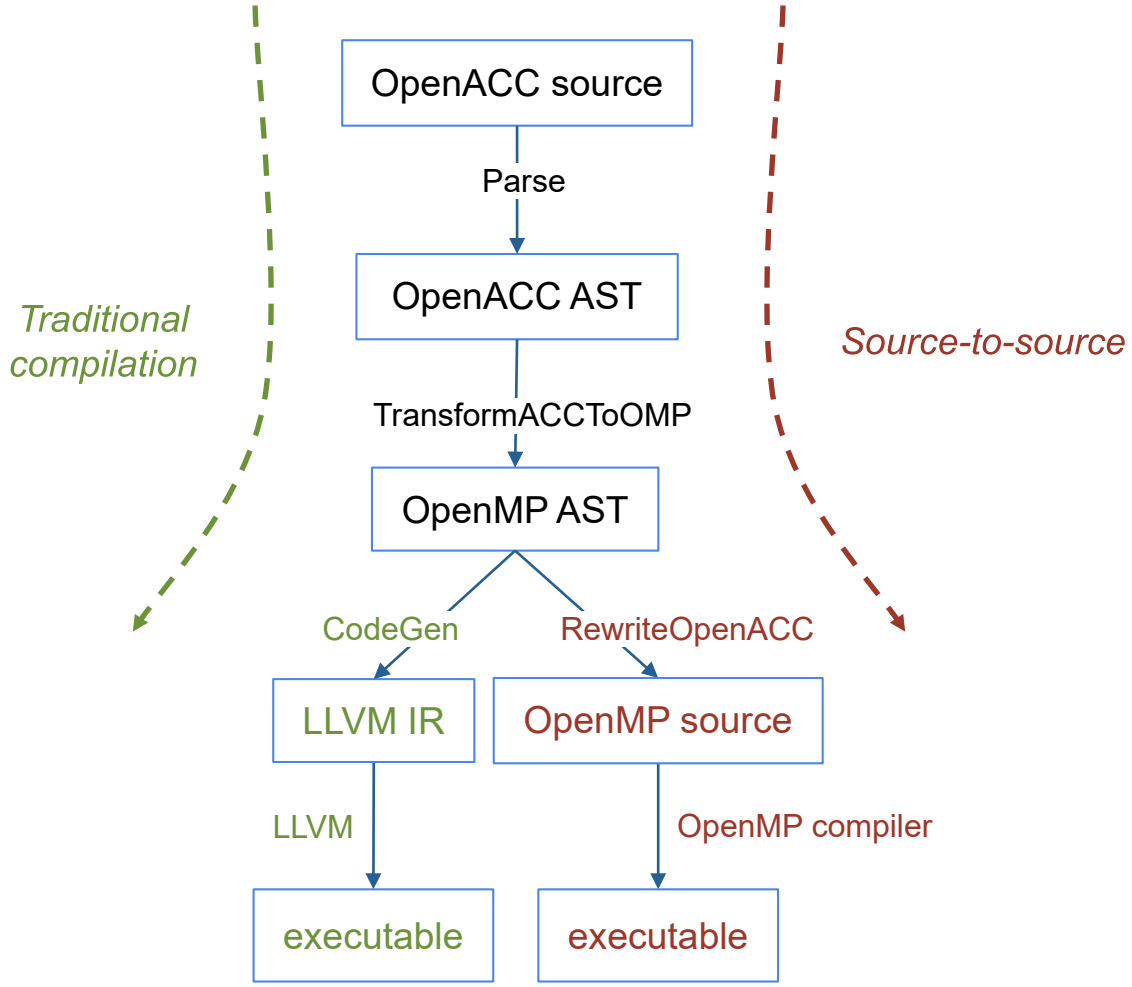
What is Clacc?

- Goal
  - OpenACC C/C++ support for Clang and LLVM
- Design
  - Translate OpenACC to OpenMP to build on OpenMP support
- Availability
  - Web page: https://csmd.ornl.gov/project/clacc
  - Source code: https://github.com/llvm-doe-org/llvm-project/wiki
- Funding
  - Exascale Computing Project (ECP)
- Contact
  - Joel E. Denny (dennyje@ornl.gov)



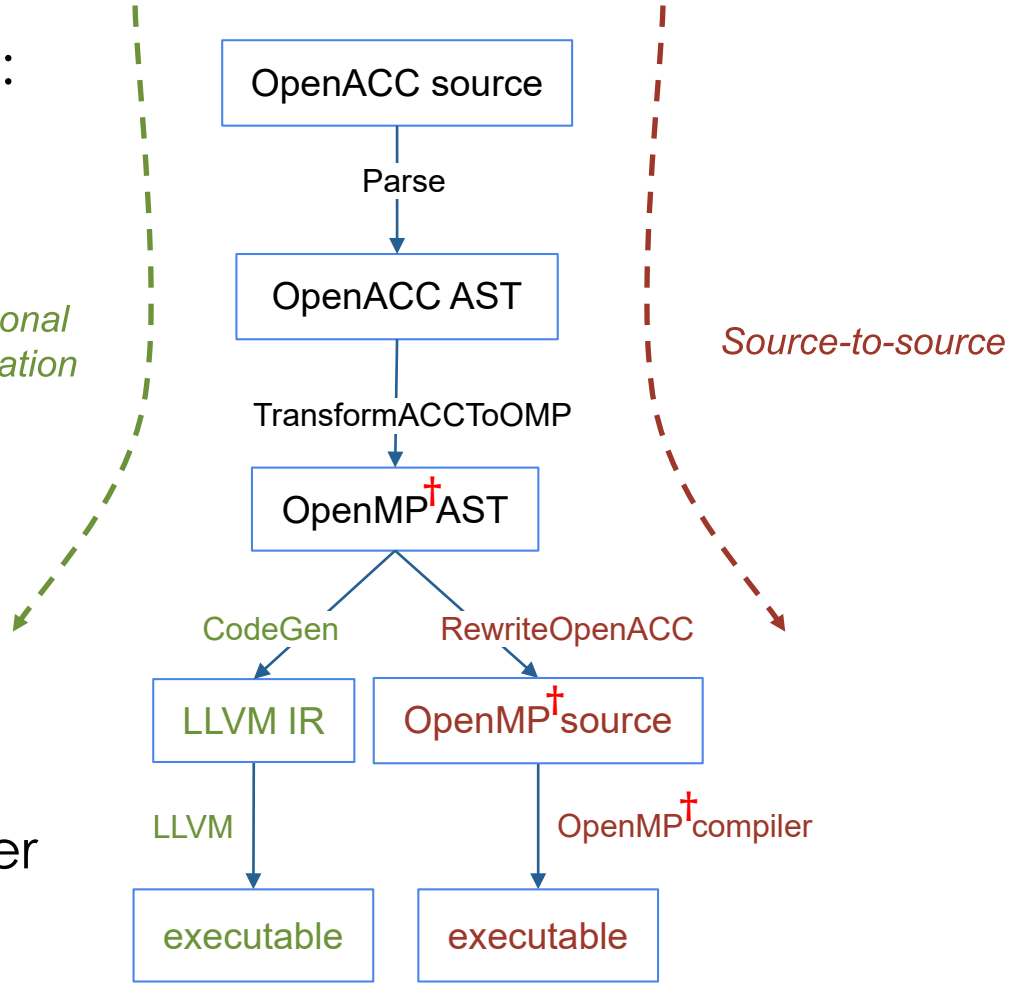
Clacc: Two Compilation Modes

- Traditional compilation
  - OpenACC source → executable
    - Similar to NVHPC or GCC
  - OpenMP serves as an internal IR
    - Diagnostics and profiling data expressed in terms of original OpenACC source not OpenMP
    - Maximizes reuse of OpenMP implementation
- Source-to-source
  - OpenACC source → OpenMP source
    - Target other OpenMP compilers and tools
    - Port apps or benchmarks
  - Uses Clang's Rewrite facility
    - Remains human-readable
    - Appropriate for other OpenMP compilers, perhaps targeting other architectures



Clacc: OpenMP Extensions†

- Problem: OpenMP can't express some features. E.g.:
  - Dual reference counters for device allocations
  - `auto` clause, `kernels` construct, `no_create` clause
- Solution: OpenMP extensions
  - Traditional compilation or source-to-source mode
  - Distinct OpenACC vs. OpenMP representations with full translation in one compiler phase
  - Complex analyses and transformation passes can be implemented on LLVM IR instead of Clang AST
  - Following OpenACC's history, leads to contributions to the OpenMP specification
- Caveat: Source-to-source vs. other OpenMP compiler
  - Compiler diagnostic if OpenMP extension used
  - Options to locate and select alternative translations



Clacc: OpenACC Runtime Library and Profiling Interface

- Again, build on OpenMP plus extensions
  - Clearly defined relationship between OpenACC and OpenMP representations
  - Provides support for source-to-source and using other OpenMP runtimes
  - Following OpenACC's history, leads to contributions to the OpenMP specification
- libacc2omp: OpenACC runtime
  - Wrapper around OpenMP runtime
  - Currently tested with LLVM's OpenMP runtime only
  - Carefully defined interfaces to facilitate extending support to other OpenMP runtimes in the future
- OpenACC Runtime Library Routines
  - libacc2omp wraps OpenMP runtime library routines plus original extensions
- OpenACC Profiling interface
  - libacc2omp wraps OpenMP's OMPT plus original extensions
- OpenACC Environment Variables
  - libacc2omp requires OpenMP runtime to call handler routines provided by libacc2omp
- Flang's Fortran OpenACC support
  - Expected to reuse Clacc's implementation of both libacc2omp and OpenMP runtime extensions

Clacc: Development Status

The most noteworthy developments from the past year are shown in **bold**

- Directives
  - Basic features are supported (e.g., data, parallel, loop directives, **acc routine seq**)
  - Some important features missing (e.g., kernels directive, C++ support, async/wait)
- **Runtime Library Routines, Preprocessor, Environment Variables**
  - **Most features supported**
  - Some missing (e.g., async/wait routines)
- Profiling interface
  - All events supported except wait events
  - Some profiling data missing (e.g., kernel_name, num_gangs, num_workers, vector_length)
  - **var_name support**
  - **Integrated with TAU**
- Supported Architectures
  - x86_64, Power 9, NVIDIA GPU
  - **AMD CPU, GPU**
- Ongoing activities
  - CI testing on ORNL's ExCL cluster and **Ascent (Summit training system)**
  - **Issue tracking on LLVM DOE Fork in github**
    - **Identifies potential external contributions**
  - Upstream Clang/LLVM
    - Merging into Clacc
    - **Contributed dual ref count support: OpenMP ompx_hold map type modifier extension**
    - Contributing various other improvements
  - Contributing improvements to OpenACC spec and OpenMP spec

Takeaways

- Clacc
  - OpenACC C/C++ support for Clang/LLVM
  - Builds on OpenMP plus extensions
  - Two compilation modes: traditional and source-to-source
- FY22 Plans
  - Develop critical missing OpenACC features
  - Upstream runtime support to share with Flang
- Join Us
  - Oak Ridge National Laboratory
  - Hiring interns, postdocs, research and technical staff
  - External collaborators welcome

- URLs
  - Web: https://csmd.ornl.gov/project/clacc
  - Source: https://github.com/llvm-doe-org/llvm-project/wiki
  - Email: dennyje@ornl.gov
- Publications
  - OpenACC Profiling Support for Clang and LLVM using Clacc and TAU, Camille Coti, Joel E. Denny, Kevin Huck, Seyong Lee, Allen D. Malony, Sameer Shende, and Jeffrey S. Vetter, ProTools, GA, USA (November 2020)
  - Clacc: Translating OpenACC to OpenMP in Clang, Joel E. Denny, Seyong Lee, and Jeffrey S. Vetter, 2018 IEEE/ACM 5th Workshop on the LLVM Compiler Infrastructure in HPC (LLVM-HPC), Dallas, TX, USA, (November 2018).