The Frontier Programming Environment at OLCF

David E. Bernholdt
Programming Environment and Tools Lead
Oak Ridge Leadership Computing Facility
Oak Ridge National Laboratory
Contributors to Frontier Programming Environment

Vendor-Provided

• Cray Programming Environment (CPE)
  – Includes Cray compiler for C, C++, and Fortran plus GCC compiler. All the Cray profiling, tuning, and debugging tools. OpenMP and Cray MPI optimized for AMD GPU direct.

• AMD ROCm programming environment
  – Includes LLVM compiler to generate optimized code for both the AMD Trento CPU and MI200 GPU. It will support: C, C++, and Fortran and have GPU offload support. HIP, a CUDA-like direct GPU programming model (with CUDA to HIP conversion utilities).

Other Sources

• ECP
  – LLVM enhancements: Flang (Fortran front-end), OpenMP, OpenACC
  – Kokkos and RAJA
  – HIP LZ (HIP support for Aurora)
  – MPI, HPCToolkit, PAPI enhancements
  – …

• ALCF + OLCF
  – Pilot implementation of DPC++/SYCL for Frontier

• OLCF
  – GCC enhancements to better support OpenACC, OpenMP, Fortran on Summit and Frontier
Programming Environment

• Compilers Offered
  – Cray PE (C/C++ LLVM-based; Cray Fortran)
  – AMD ROCm (LLVM-based)
  – GCC

• Programming Languages & Models Supported (in which compilers)
  – C, C++, Fortran (all)
  – OpenACC (GCC) 2.6 substantially complete, 2.7 planned
  – OpenMP (all) 5.0-5.2 in progress – most priority features complete, details vary
  – HIP (Cray, AMD)
  – Kokkos/RAJA (all)
  – UPC (Cray, GCC)

• Transition Paths
  – CUDA: semi-automatic translation to HIP
  – CUDA Fortran: HIP kernels called from Fortran (a more portable approach)
    o CUDA Fortran kernels need to be translated to C++/HIP (manual process)
    o Fortran bindings to HIP and ROCm libraries and HIP runtime available through AMD’s hipfort project
# Programming Tools

## Debuggers and Correctness Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>System-Level Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arm DDT</td>
<td></td>
</tr>
<tr>
<td>Cray CCDB</td>
<td></td>
</tr>
<tr>
<td>Cray ATP</td>
<td></td>
</tr>
<tr>
<td>STAT</td>
<td></td>
</tr>
<tr>
<td><strong>Node-Level Tools</strong></td>
<td></td>
</tr>
<tr>
<td>ROCgdb</td>
<td></td>
</tr>
<tr>
<td>Cray GDB4HPC</td>
<td></td>
</tr>
</tbody>
</table>

## Performance Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>System-Level Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arm MAP/Performance Reports</td>
<td></td>
</tr>
<tr>
<td>CrayPat/Apprentice2 (Cray)</td>
<td></td>
</tr>
<tr>
<td>Reveal (Cray)</td>
<td></td>
</tr>
<tr>
<td>TAU</td>
<td></td>
</tr>
<tr>
<td>HPCToolkit</td>
<td></td>
</tr>
<tr>
<td>Score-P / VAMPIR</td>
<td></td>
</tr>
<tr>
<td><strong>Node-Level Tools</strong></td>
<td></td>
</tr>
<tr>
<td>gprof</td>
<td></td>
</tr>
<tr>
<td>PAPI</td>
<td></td>
</tr>
<tr>
<td>ROCprof</td>
<td></td>
</tr>
<tr>
<td>ROC-profiler &amp; ROC-tracer libraries</td>
<td></td>
</tr>
</tbody>
</table>

Items in green are also available on Summit
# Scientific Libraries and Tools

<table>
<thead>
<tr>
<th>Functionality</th>
<th>CPU</th>
<th>GPU</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLAS</td>
<td>Cray LibSci, AMD BLIS, PLASMA</td>
<td>Cray LibSci_ACC, AMD roc/hipBLAS, AMD rocAMD ROCm Tensile, MAGMA</td>
<td>MAGMA and PLASMA are open source software led by the UTK Innovative Computing Laboratory</td>
</tr>
<tr>
<td>LAPACK</td>
<td>Cray LibSci, AMD libFlame, PLASMA</td>
<td>Cray LibSci_ACC, AMD roc/hipSolver, MAGMA</td>
<td></td>
</tr>
<tr>
<td>ScaLAPACK</td>
<td>Cray LibSci</td>
<td>ECP SLATE, Cray LibSci_ACC</td>
<td></td>
</tr>
<tr>
<td>Sparse</td>
<td></td>
<td>AMD roc/hipSparse, AMD rocALUTION</td>
<td></td>
</tr>
<tr>
<td>Mixed-precision iterative refinement</td>
<td>Cray IRT, MAGMA</td>
<td>MAGMA</td>
<td></td>
</tr>
<tr>
<td>FFTW or similar</td>
<td>Cray, AMD, ECP FFTX, FFT-ECP</td>
<td>AMD rocFFT, ECP FFTX, FFT-ECP</td>
<td>FFT-ECP focuses on 3D FFTs</td>
</tr>
<tr>
<td>PETSc, Trilinos, HYPRE, SUNDIALS, SuperLU</td>
<td></td>
<td></td>
<td>Spack recipes from ECP xSDK</td>
</tr>
</tbody>
</table>

Functionality in green is also available on Summit
Digging a Little Deeper
For C/C++ Codes

• Multiple compilers available
  – AMD
  – Cray
  – LLVM

• But they’re all based on LLVM
  – HPE and AMD are among the many organizations contributing to the development of LLVM
  – Most work is “upstreamed” (contributed to the core LLVM source)
    o But not everything is accepted (immediately), or may be held back as proprietary
  – Capabilities (and bugs) are likely to be generally similar at any point in time…
  – But not identical!

• LLVM is also available on Summit
For Fortran Codes

• One useful compiler available at present
  – Cray
    o *Not* based on LLVM

• AMD provides a Fortran implementation, but we don’t recommend it
  – It is based on “classic Flang”, in the LLVM ecosystem
  – Support for both the latest language standards and OpenMP offload are limited

• There is extensive work underway in the LLVM community on Flang, but it will be some time before it is production quality
But What About GCC?

- On this slide “gcc” refers to the whole suite, including gfortran
- OLCF is working with Siemens to implement OpenMP in gcc
- OLCF will provide recent release and development versions of gcc on Frontier
- For various reasons, you should not expect gcc-generated executables to be performant for offload at this time
  - Results will vary
  - We are interested in improving the performance of gcc. If you have a troublesome case, reach out to me. (No guarantees, however)
- GCC is also available on Summit
For HIP (and CUDA) Codes

- HIP runs today on AMD and NVIDIA GPUs
- An ECP project is working on supporting HIP on Intel GPUs too
- Recommend a one-time translation of CUDA codes to HIP and make the HIP version primary from then on
- Both Cray and AMD compilers support HIP
  - They both use the AMD runtime
- More on HIP in the next two tutorials in this series
- HIP is also available on Summit
For OpenMP Codes

• OpenMP is very much a work in progress in the LLVM community
  – Most of 5.0 is implemented
  – Parts of 5.1, 5.2 are implemented

• We (DOE labs, including ORNL/OLCF) are trying to help prioritize the order of implementation based on what users tell us they need/want
  – So if you could really use features that aren’t available yet, please let us know!

• Cray and AMD compilers use different OpenMP runtimes

• Remember that Cray Fortran is not based on LLVM

• OpenMP implementation in gcc is also a work in progress

• More on OpenMP later in this series

• OpenMP is available on Summit, but different progress on impl
For OpenACC Codes

• OLCF provides OpenACC support via gcc
  – 2.6 currently supported
  – 2.7 planned
  – 3.x not currently planned – let us know if there are particular features that you could really use
  – Don’t expect this to be performant at present

• Cray Fortran supports OpenACC 2.0
  – Work is underway to 3.2 (latest) but no timeline has been given
  – No OpenACC in Cray C/C++

• Work is also underway in the LLVM community on OpenACC
  – Unknown when these will be production

• OpenACC is also available on Summit
What about SYCL?

• OLCF and ALCF have partnered with Codeplay on a pilot implementation of the Intel DPC++ compiler for AMD GPUs
  – ALCF has also partnered with NERSC on NVIDIA support

• Pilot implementation is complete
  – ~“50%” level of support
  – Tested with a small set of benchmarks and mini-apps
  – Should be available any day now on Spock, with Crusher/Frontier soon to follow

• Seeking interested users to try out the pilot implementation
  – Provide feedback
  – Shake out issues
  – Provide motivation to complete the port

• Can make SYCL available on Summit if there is interest
Help Us Help You…

• If you have a liaison, work with them

• **If you encounter an issue, file a ticket with OLCF** – otherwise the facility won’t (necessarily) know about it, and can’t track it
  – Summit, Spock, Crusher, Frontier…

• Take advantage of training events
  – Future events in the *Preparing for Frontier* series
  – If you have Crusher access: office hours, hackathons, additional trainings